Image Cover Sheet

CLASSIFICATION	SYSTEM NUMBER 515910
UNCLASSIFIED	

TITLE

CF-18 structural life analysis model: a user's guide

System Number:

Patron Number:

Requester:

Notes:

DSIS Use only:

Deliver to: DK

This page is left blank

This page is left blank

DEPARTMENT OF NATIONAL DEFENCE CANADA
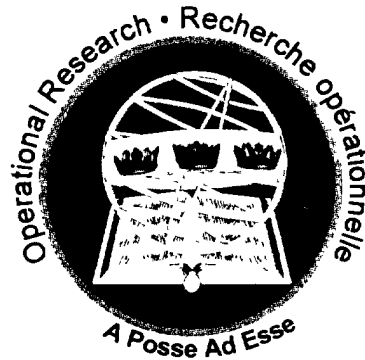
OPERATIONAL RESEARCH DIVISION

DIRECTORATE OF OPERATIONAL RESEARCH (MARITIME, LAND & AIR)

ORD PROJECT REPORT PR 2001/08

CF-18 STRUCTURAL LIFE ANALYSIS MODEL: A USER'S GUIDE

by

LUMINITA STEMATE

JUNE 2001

OTTAWA, CANADA

National Défense
Defence nationale

OPERATIONAL RESEARCH DIVISION

CATEGORIES OF PUBLICATION

ORD Reports are the most authoritative and most carefully considered publications of the DGOR scientific community. They normally embody the results of major research activities or are significant works of lasting value or provide a comprehensive view on major defence research initiatives. ORD Reports are approved personally by DGOR, and are subject to peer review.

ORD Project Reports record the analysis and results of studies conducted for specific sponsors. This Category is the main vehicle to report completed research to the sponsors and may also describe a significant milestone in ongoing work. They are approved by DGOR and are subject to peer review. They are released initially to sponsors and may, with sponsor approval, be released to other agencies having an interest in the material.

Directorate Research Notes are issued by directorates. They are intended to outline, develop or document proposals, ideas, analysis or models which do not warrant more formal publication. They may record development work done in support of sponsored projects which could be applied elsewhere in the future. As such they help serve as the corporate scientific memory of the directorates.

ORD Journal Reprints provide readily available copies of articles published with DGOR approval, by OR researchers in learned journals, open technical publications, proceedings, etc.

ORD Contractor Reports document research done under contract of DGOR agencies by industrial concerns, universities, consultants, other government departments or agencies, etc. The scientific content is the responsibility of the originator but has been reviewed by the scientific authority for the contract and approved for release by DGOR.

DEPARTMENT OF NATIONAL DEFENCE

CANADA

OPERATIONAL RESEARCH DIVISION

DIRECTORATE OF OPERATIONAL RESEARCH
(MARITIME, LAND & AIR)

ORD PROJECT REPORT 2001/08

CF-18 STRUCTURAL LIFE ANALYSIS MODEL:
A USER'S GUIDE

by

-

LUMINITA STEMATE

Recommended by: J. EVANS   Approved by: A. BRADFIELD
                   DOR(MLA)   DGOR

OTTAWA, ONTARIO   JUNE 2001

ABSTRACT

This study represents a comprehensive *user's guide* for the 2000 version of the CF-18 Structural Life Analysis Model (CF-18 SLAM) that was originally developed in 1989 and has been used ever since as a support tool for various decisions regarding the structural life of the CF-18 fleet. This guide offers a high-level, "philosophical" perspective on the model, as well as a low-level, programming perspective, the whole complemented by a suite of discussed examples.

RÉSUMÉ

Cette étude représente un *guide de l'usager* complet, réalisé pour la version 2000 du modèle d'analyse de la vie structurelle des aéronefs CF-18, dont la version originale a été conçue en 1989 et qui depuis a été continuellement utilisé comme aide pour la prise de décisions regardant la vie structurelle de cette flotte d'aéronefs. Le guide offre une vue d'ensemble, "philosophique", du modèle, ainsi qu'une vue approfondie de détails de programmation, le tout complété par une série d'exemples commentés.

ACKNOWLEDGEMENTS

EXECUTIVE SUMMARY

1.   The CF-18 Structural Life Analysis Model (SLAM) was originally developed in 1989 and has been used ever since as a support tool for various decisions regarding the structural life of the CF-18 fleet. Since its initial development, the model underwent numerous modifications to keep it current with the changes occurring in the CF-18 fleet composition and mode of operation.

2.   The main purpose of the present study is to provide a comprehensive description of the latest version of the CF-18 SLAM, under the form of a user's guide. The guide contains a thorough description of the algorithms involved in the model, along with a programming perspective of the same algorithms, the whole completed by a set of discussed examples of analyses that were performed using the model.   This document can be useful to three types of audiences:

   a) the analyst called to use the present version of the model;

   b) the analyst called to either modify the present version of the model or to develop a new model of the same type; and

   c) the CF-18 community.

3.   Users of type a) can benefit from every part of this document. The description of the algorithms will help them gain a deeper understanding of the philosophy of the model, most necessary in order to be able to make full use of the model.   The programming details provided will help them use the present implementation of the model to implement new scenarios, as required.   Finally, the included examples of analyses will help introduce them to the panoply of output results that can be obtained through the model.

4.   Users of type b) will benefit mostly from the description of the algorithms from a high-level, "philosophical" perspective. The set of discussed examples of analyses can be of some assistance to this type of user, as well.

5   The CF-18 community will be interested in the latest results that have been obtained through the model. These can be found in the second part of the report.

6.   The result considered by the author of utmost importance, concerns the *availability* of CF-18 aircraft in the years to come, as a consequence of the present program of upkeep and modernization of 80 aircraft (Figure A).
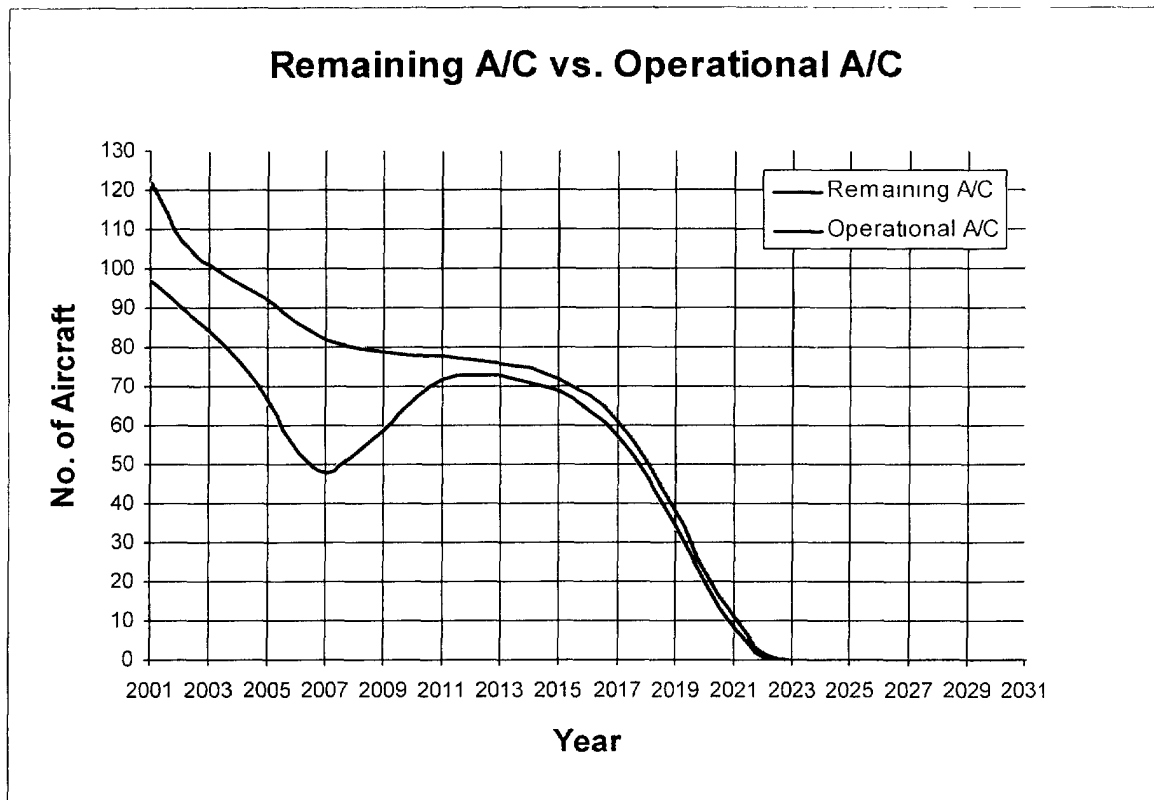
Figure A. Number of remaining aircraft vs. number of operational aircraft in the baseline scenario

7.   Figure A. distinguishes between the expected number of *remaining* aircraft in each of the following years and the expected number of *operational* aircraft during the same period.   The "operational" aircraft are defined as aircraft that are available for flying (i.e., not crashed, not fatigued-out, not in second or third line maintenance and not waiting to be accepted in plant for third line maintenance activities).   From the difference

between the profiles of the two curves, it can be said that it would be risky to use the number of remaining aircraft as the main parameter used for planning purposes, especially concerning the planning for training activities. The number of operational aircraft is a much more adequate measure and its use for planning purposes is strongly recommended.

TABLE OF CONTENTS

CF-18 STRUCTURAL LIFE ANALYSIS MODEL

USER'S GUIDE

INTRODUCTION

1.   The present text is intended to accompany the C++ version of the CF-18 Structural Life Analysis Model (CF-18 SLAM). The model presented here represents the most recent update of the original Fortran model developed in 1989 (see Ref. 1). Between the first and the present version of the model, there were several updates and refinements of the CF-18 SLAM (see Ref. 2 and Ref. 3). All these references are strongly suggested as a complementary source of information. Although the philosophy of the original model has been preserved, an important number of modifications have been made, mainly in order to reflect the current status and concept of operations of the CF-18 fleet. The version described here is current as of March 2001.

2.   The guide has two main objectives. The first is to offer a general presentation of the model, including all the algorithms involved, along with a programming perspective of the same algorithms. The second is to show, by means of discussed examples, some of the results that can be obtained, thus allowing a user to implement their own scenarios and make full use of the model.

3.   According to the two stated objectives, the guide has been organized in two parts. The first part contains a detailed description of all the algorithms involved, most necessary for a full comprehension of the model's capabilities and limitations. Detailed descriptions of all the functions involved in the implementation are also provided.   In the second part, the focus is on how the model can be used. Several scenarios are considered and the model's inputs and outputs are discussed in detail.

- 2 -

PART I: A DETAILED DESCRIPTION OF THE SIMULATION MODEL

SIMULATION METHODOLOGY

4.   The model is intended to reflect (simulate) all major features of the CF-18 fleet, which, along with the model's forecasting capabilities, will provide a powerful tool for supporting senior decisions regarding the management of the fleet. Aircraft are treated individually and most of their attributes are recognized in the simulation. Among these attributes are: the type of the aircraft (single or dual), the current number of hours flown as well as the current value of the Fatigue Life Expended Index (FLEI) or the life limit of each aircraft (i.e., if it is an early production aircraft or not). Analysis and forecasting are based on a statistical approach using mean values and standard deviations obtained from historical data.

5.   The implementation is done using a "baseline scenario" consisting of a certain number of assumptions that were valid at the moment the model was last modified (i.e., March 2001). These assumptions will be discussed further, each of them in the context of the appropriate algorithm. An analyst with average programming knowledge will be able to implement almost any given scenario, by modifying the baseline scenario already implemented.

6.   In an attempt to express the philosophy of the model in one phrase, this would be: "*In a probabilistic manner, the CF-18 SLAM simulates what could happen to any particular aircraft in any particular year during the simulation horizon.*" Several iterations will provide us with the necessary data to calculate the mean values for all our parameters of interest, such as the number of remaining aircraft in any year, etc.

7.   The main events that are modeled can actually be introduced as answers to the question: "What could happen to any particular aircraft in any particular year?" Thus, we have:

   a)   an aircraft could fly (Functions associated: "*hours_flown*", "*allocate_fatigue*");

- 3 -

    b) an aircraft could undergo maintenance activities (Functions associated: *"second_line_maintenance"*, *"third_line_maintenance"*);

    c) an aircraft could crash (Functions associated: *"crash_algorithm"*, *"update_crash_data"*); and

    d) an aircraft could fatigue out (Function associated: *"fatigued_out_algorithm"*).

8.   These functions constitute the main building blocks of the model. They will be referred to in the present text as "key functions". However, the model implements several other functions, such as the one used to read the input data or the one used to compile and store the results. They will be referred to as "auxiliary functions". Both types will be thoroughly described. The first group of functions will provide the user with a high-level, philosophical perspective of the model. The second group of functions will facilitate a good understanding of the model from a low-level, programming perspective.

KEY FUNCTIONS / EVENTS

Attrition

9.   Peacetime accidents (crashes) are mainly of three categories: (1) due to human errors, (2) engine related, and (3) due to unknown causes. In Canada, in 12 out of 15 losses experienced up to date by the CF-18 fleet, (i.e., in 80 % of the losses), personnel (not necessarily the pilot) was the primary cause. These statistics indicate that an attrition model based on personnel issues such as the level of experience would be the most appropriate. The Rand "learning curve" model was selected in the original version of the simulation (see Ref. 2) and the same model has been kept in the present version, with slight modifications at the implementation level.

10.   According to the Rand model, the attrition rate is decreasing in time, mainly because the personnel are gaining more experience and a better knowledge of the aircraft. The equation describing this curve is the following:

- 4 -

$$L = ah^b \qquad (1)$$

where L represents the cumulative number of attrition losses. h is the cumulative flying hours (for the whole fleet) and a and b are constants. The constants were initially fitted based on the original aircraft attrition forecasts, and they are continually updated. They are actually updated each month such that the last point on the curve corresponds to the number of hours flown up to date. The monthly variations are minor. More important variations could occur after a new crash. A methodology for deciding whether a new crash should be considered in adjusting the parameters of the curve was described in an earlier study (see Ref. 4).

11.   However, it could be argued that, independent of the level of experience gained by the personnel, there would always be accidents that are due to materiel failure. bird strikes, or other non-human related causes. Are these accounted for in the Rand model or not? Or, in other words, is the Rand learning curve fully satisfactory for the purpose of modeling peacetime aircraft attrition? The following discussion attempts to answer this question.

12.   The Rand learning curve is very reliable for short-term predictions. and in the short term, it actually accounts for both human and non-human related accidents, because *all* accidents are included to find the best curve-fit. However, it becomes gradually less reliable as we advance our predictions into the future, because late into the future the curve tends to represent only the 80 % of the accidents that are due to human errors. It assumes that the attrition rate will continually decrease (due to the learning process) going asymptotically to zero. whereas this is practically impossible because of that portion of the accidents that are independent of any learning process. This being said. this would represent a weakness of the model only if equation (1) was used in a deterministic[1] way. But L should be regarded as an average value and, consequently, in the implementation of the model. some variation has to be added to this average value. It

---

[1] "Deterministic", in the sense that, given the present values for a and b and a cumulative number of flying hours for the fleet, the cumulative number of losses corresponding to that number of flying hours would be determined using the formula (1).

- 5 -

is the author's opinion that the randomness resulting this way offers a good coverage of the non-human related accidents' zone, the Rand learning curve thus becoming a reasonably satisfactory peacetime attrition model.

13.   The function that implements this attrition model is called "*crash_algorithm*" and it returns the number of crashes occurring in the current year. The Rand learning curve provides us with a mean value for the number of losses that might be expected in each particular year in the future. As it was explained in the previous paragraph, for the purposes of our simulation, a distribution is needed to describe the random variations that can occur around this mean value. The Poisson distribution has been chosen for this purpose. The formula used is the following:

$$P(n) = \frac{(Rh)^n e^{-Rh}}{n!} \tag{2}$$

where *P(n)* represents the probability of n accidents occurring in a subinterval of one year ($= h$ = YFR (Yearly Flying Rate)), while Rh represents the expected number of losses within the interval of one year (the current year) and is calculated according to the formula:

$$Rh = L_2 - L_1 \tag{3}$$

where L_2 is the expected number of losses at the end of the current year and L_1 is the expected number of losses at the beginning of the current year (i.e., at the end of the previous year).

Therefore,

$$Rh = ah_2^b - ah_1^b \tag{4}$$

where h_2 represents the cumulative number of flying hours for the fleet at the end of the current year, while h_1 represents the cumulative number of flying hours for the fleet at the end of the previous year (i.e., $h_2 = h_1$ + YFR).

- 6 -

14.   Associated with the function *"crash_algorithm"* (determining the number of crashes occurring in the current year) is the function *"update_crash_data"*, which is used to update the database after the eventual crashes. More precisely, the second function has, as input parameter, the number of crashes (which is returned by the first function) and must pick randomly, from all aircraft that have flown sometime during that year (and therefore had a "chance" to crash), the required number of aircraft. The chosen aircraft will have the field describing if they have crashed or not, changed from a value of "NO" to "YES".

Maintenance activities

15.   There are several kinds of maintenance activities, classified in three groups:

a) 1st line maintenance. Includes minor repairs as well as inspections of the aircraft before and after each flight; they are effectuated at each squadron's location. The duration of 1st line maintenance activities may be anywhere between a few hours and a few days;

b) 2nd line maintenance. Includes the periodic inspections that are conducted on the aircraft after each 400 flying hours. These inspections are also effectuated at the squadron's location and they usually take approximately one month to execute;

c) 3rd line maintenance. Includes any structural modifications (or major repairs, but these are rare) that are to be done on the aircraft. They are conducted in plant (Bombardier) and usually take several months, depending on the complexity of the set of modifications that is implemented. Upgrades in avionics are also included here.

16.   Given the duration of each of the three groups of maintenance activities, it follows that only the last two groups have a significant influence in maintaining the operational effectiveness of the fleet. Therefore, the present model implements two separate functions (bearing self-descriptive names) to simulate these maintenance activities: *"second_line_maintenance"* and *"third_line_maintenance"*.

- 7 -

17.   The *second_line_maintenance* function scans the number of hours flown by each aircraft and decides, each month, which aircraft are due for second line maintenance. All aircraft are eligible to go into second line maintenance, as long as they fly. The algorithm governing this function is very simple: if the difference between the number of hours flown by aircraft *j* up to the current month and the number of hours flown by aircraft *j* when the aircraft had undergone its last periodic inspection is greater or equal to 400 hours, then the aircraft is sent for inspection. This function also counts the number of aircraft that are sent into second line maintenance in the current month.

18.   The *third_line_maintenance* function implements the structural modification programs to be done at Bombardier. At the Force Structure Exercise (FSX) 2000, it has been decided that there is a requirement to upkeep and modernize 80 aircraft (from the total of 122 actually in the fleet). The modernization, consisting of avionics upgrades, will be done under the CF-18 Incremental Modernization Program (IMP). The date of the implementation of this program is not known presently. On the other hand, the implementation of the structural modifications program has already begun. The structural modifications are required to increase the life limit of the aircraft up to a FLEI[2] of 1.0. The following paragraphs give a more detailed description of this program, on which the function "*third_line_maintenance*" is based. A description of the function itself is also given.

19.   The International Follow-On Structural Test Project (IFOSTP) determined that CF-18 aircraft should be removed from flight status once they reach a 0.52 FLEI structural threshold, to prevent irreversible damage of one of the bulkheads. Once this threshold is reached, two choices are available: (1) to retire the aircraft and declare it as "surplus aircraft", to be used either for sale, spare parts or as a war reserve, or (2) to upkeep and modernize the aircraft. The 0.52 threshold has to do with financial risk, in the sense that there is a high risk for aircraft that have passed this threshold to experience irreversible damage to the bulkhead. For the aircraft selected not to be part of the

---

[2] See pages 11-12 for more information on FLEI.

- 8 -

modernization program, additional financial risk has been accepted and a new threshold has been set at 0.56. Contrary to the 0.52 threshold, the 0.56 one is airworthiness related and cannot be exceeded.

20.   This additional financial risk was not accepted for the group of aircraft selected to be modernized. Moreover, recognizing the fact that the 0.52 threshold is not 100 % reliable, a more conservative one will be actually used in practice: 0.515. Once this threshold is reached, an aircraft will have to undergo important structural modifications in order for its FLEI to be extended to 1.0. For financial reasons (the sum required is not available in total from the very beginning, and only a limited budget is available each year) as well as because of the large number of modifications to be executed, it has been decided to divide the package of structural modifications into three smaller ones. They will be referred to as CP1, CP2 and CP3 (standing for "Control Point" 1, 2 and 3). The modifications included in the first package (CP1) will result in an extension of the fatigue life up to a FLEI of 0.645. The second package (CP2) will extend the fatigue life up to a FLEI of 0.733, while the third package (CP3) will further extend the life of the aircraft up to a FLEI of 1.0. Presently, the development, from an engineering point of view, of the modifications included in CP1 is completed. CP2 and CP3 are at different stages of development, CP2 being expected to be finished by 2002, while CP3 is expected to be completely developed by 2003.

21.   Given that the processes of dismantling and re-assembling the aircraft, necessary to execute any package of structural modifications as well as avionics upgrades, are quite resource intensive, and therefore expensive, there is a requirement to reduce their number as much as possible. For this reason, it has been decided that, even if an aircraft has to undergo three packages of modifications, they will have to be done in at most two passes through Bombardier. The combinations that have been retained are the following:

a)   aircraft will enter the plant for CP1, then fly until the threshold required for CP2 is attained and then return to Bombardier for CP2 and CP3 combined (two passes);

- 9 -

    b) aircraft will receive CP1 and CP2 combined when they reach a FLEI of 0.52, then fly until they reach a threshold of 0.733, at which point they return to Bombardier to receive CP3 (two passes);

    c) aircraft will receive CP1, CP2 and CP3 combined when they reach a threshold of 0.52 (one pass);

These are the combinations agreed upon between DND and Bombardier and they apply to the majority of the 80 aircraft selected. However, four of these aircraft will undergo a different program, the replacement of the centre barrel. These aircraft tail numbers' are 747, 749, 758 and 759.

22.   In what concerns the Incremental Modernization Program, very little details are presently known. Phase I of this program is estimated to begin by mid 2002 and last until 2006, when Phase II will commence. The entire program is estimated to last until 2008 (see Ref. 6). IMP will be done concurrently with the structural modifications program. Obviously, the time that the aircraft will have to spend in plant undergoing both types of modifications (structural and avionics) will be longer than if the aircraft was there solely for the structural modifications. However, since no details are presently available, modeling the IMP was not possible.

23.   The function "*third_line_maintenance*" implements the maintenance process described above. First, a test is performed in order to identify the aircraft that are eligible to undergo the three sets of modifications (CP1, CP2 and CP3). The reader is reminded that only 80 aircraft from the present 122 of the fleet are eligible for upkeep and modernization. The test uses the list of aircraft selected for early retirement, as specified by the user. The function then, gives priority to the aircraft that are in a "waiting" status, to enter the plant, provided that the working capacity available at Bombardier is sufficient. In the current implementation (March 2001), it is considered that Bombardier has enough resources to work on 12 aircraft in any given month[3]. For each aircraft

---

[3] This is based on the induction plan provided by Bombardier for the period September 2000 to January 2003. According to this induction plan, the average number of aircraft in the plant in any given month is 11 5.

- 10 -

waiting, the function determines the package of modifications the aircraft is supposed to undergo, according to its current FLEI level and to the set of modifications already received by the aircraft (if any).

24.   In case there are no aircraft waiting to enter the plant for 3rd line maintenance activities, or after these aircraft have been taken care of, the function scans all eligible aircraft with the aim of assigning a combination of modifications (from the three combinations defined previously), to each aircraft. In addition, the function also attempts to send to the plant any aircraft that are due for one of the modification packages. This attempt may be successful or not, depending if the maximum intake level at Bombardier has been reached for the current month. If an aircraft can enter the plant, it will be flagged as being in 3rd line maintenance for a specified number of months, according to the duration of the package of modifications that the aircraft will undergo. If it cannot enter the plant (because of the limited capacity existent at Bombardier), the aircraft will be flagged as "waiting". When in "waiting" status, the aircraft cannot fly until it receives the required structural modifications. This function also counts both the number of aircraft in plant in the current month and the number of aircraft waiting to be sent in plant (also in the current month).

25.   It is important to note that this function requires updating. At this stage, the author can only point out the reasons for which an update will be required, leaving the "when" and "how" to the judgement of the analyst in charge of the model. These reasons include the following:

a) Some of the assumptions may lose their validity. For example, now it is believed that the second package of modifications, CP2, will be available in 2002; should future developments show that it will actually become available in 2003, the function would have to be modified, in order to account for this change;

b) Even in the eventuality that all assumptions keep their validity, special attention should be directed towards the ones that are time related. For example, suppose that CP2 will indeed become available in 2002 (i.e., the assumption made today (March 2001) will remain valid in 2002). In the language of the model, the year

- 11 -

"2002" will be described differently in a simulation conducted in 2001 than in a simulation conducted in 2002. In the first case, "2002" will be the second year of the simulation (described as "*year* = 1" in the program), while in the second case, "2002" will become the first year of the simulation (described as "*year* = 0" in the program). This type of updates should be done yearly; and

c) As soon as information becomes available concerning the Incremental Modernization Program, this function would have to be modified to incorporate any new knowledge.

Fatigue

26.   Fatigue occurs when fluctuating loads are applied to the structure. It represents one of the major factors compromising the structural integrity of an airframe. The measure used to quantify the fatigue damage accumulated by an aircraft is the Fatigue Life Expended Index[4] (FLEI). The instrument used to measure FLEI is an accelerometer that adds up the number of 'g' counts that it experiences in its location near the centre of gravity of the aircraft or at the wing root. Then, the fatigue formula for the aircraft turns the 'g' counts during a flight into the FLEI consumed. There are two major problems in monitoring the fatigue. The first one is that the fatigue depends on many more factors than the 'g' counts and therefore it cannot be accurately measured by the system currently in place. The second problem is the large scatter in the number of cycles to failure that result when nominally identical specimens are subjected to the same loading spectrum.

27.   As most military aircraft, the CF-18 fleet is governed by a "Safe Life" philosophy which, in opposition to the "Fail Safe" philosophy (used more for transport aircraft) is aiming to design a structure that will be essentially free of cracks for the required operational life. According to this philosophy, a *scatter factor* between three and five must be applied in the testing process of a complete airframe (i.e., the specimen tested must "survive" at least 3 times the required operational life).

- 12 -

28.   Originally, the CF-18 was designed for an operational life of 6000 hours. To obtain such a life, the manufacturer, McDonnell Douglas Aircraft Company (McAir) used a scatter factor of 2. The CF found this value inadequate and instead applied a scatter factor of 3. Consequently, the majority of the CF-18s were considered to be limited to a maximum life of 4000 hours, corresponding to an FLEI of 0.667. The early production aircraft were considered to be limited to a maximum life of 2000 hours, or a FLEI of 0.333. However, recently it has been discovered (during the IFOSTP testing), that not even these limits are achievable without significant structural modifications. The new threshold, airworthiness related, as determined by IFOSTP is 0.56.

29.   A "fatigued-out" aircraft can be defined in two ways: in terms of its FLEI or in terms of its "equivalent flying hours" (EFH). The equivalence between the two definitions, both using measures of the fatigue accumulated by an aircraft, is straightforward: the design value of 6000 EFH corresponds to an FLEI of 1. The equation correlating the two is (for an aircraft j):

$$EFH(j) = FLE(j) * 6000 \qquad (5)$$

In the present model, the measure used to determine whether an aircraft is fatigued-out or not is the FLEI. More precisely, an aircraft is declared as "fatigued-out" when the FLEI has attained its upper limit. The upper limit is 1.0 for the aircraft selected for upkeep and modernization and 0.56 for most of the aircraft that have been declared "surplus" and are not already retired. In the program, three functions are directly dealing with fatigue: "allocate_fatigue", "update_FLEI" and "fatigued_out_algorithm".

30.   The first function, "allocate_fatigue", assigns a certain fatigue rate to each aircraft that has flown in the current year. Acknowledging the fact that not all aircraft are subject to the same fatigue rate, the assignment process is based on the assumption that fatigue rates are normally distributed. The mean value of this distribution is established by doctrine documents while the standard deviation has been determined from historical

---

[4] FLEI = The amount of fatigue damage accumulated as a fraction of the total structural fatigue life defined by full scale testing

- 13 -

data. The fatigue rate is assigned at the beginning of the year, and remains constant (for a given aircraft) throughout the year.

31. The second function, *"update_FLEI"* calculates, according to the fatigue rate attributed to an aircraft by the function *"allocate_fatigue"* and to the number of hours flown attributed to the aircraft by the function *"hours_flown"*, the fatigue gained in the current month by the aircraft in question. It also updates the database with the current values of FLEI.

32. The third function, *"fatigued_out_algorithm"* has as its main role to identify the aircraft that have fatigued-out during the current year. The current FLEI value is compared, for each aircraft, to the threshold corresponding to that particular aircraft. For the group of aircraft selected for upkeep and modernization, this threshold is 1.0. For the other group of aircraft (declared as "surplus"), the thresholds used are those indicated by the user (0.56 for most of the aircraft, but some of them have different life limits). Finally, this function also counts the number of aircraft that have fatigued-out during the current year, as well as the number of aircraft remaining in the current year.

33. In an older version of the program (see Ref. 3), there was another function that was related to the fatigue. More precisely, this function dealt with the FLEI dispersion and was used to achieve a better management of the fleet. The idea was to classify the aircraft into four colour coded groups (green, yellow, red and black), depending on their FLEI, so that aircraft with less fatigue accumulated will be identified as such and can be flown "harder" than those with more fatigue accumulated, with the final objective of all aircraft becoming ready for retirement approximately at the same time. This function was used to prove the need of a better management of the fleet. However, once this message was conveyed and the colour code scheme implemented by the operational CF-18 community, the function was not used later on for simulation purposes. Hence, it has not been included in the present version of the program. But it is a resource that exists and, should the need for it arise sometime in the future, it could easily be added to the model. To this purpose, the cited document (Ref. 3) should be used, keeping in mind that

- 14 -

the equations given in the reference have to be updated in order to extend the time horizon beyond 2015, which is now required.

Flying Hours

34.   The "Yearly Flying Rate" (YFR) is one of the most important and critical parameters involved in the simulation model. The YFR is a pre-approved annual value indicating how much the aircraft in the fleet are allowed to fly in the current year. It is dependent on a combination of factors such as the training requirements and the budget available. In the model, the function *"hours_flown"* allocates flying hours to the aircraft, on a monthly basis, in such a way that the sum of the hours allocated to all aircraft in one year is as close as possible to the relevant YFR. Moreover, the process of allocating hours must capture as many characteristics of the real life flying process as possible. The most important feature that is captured by this function is the fact that the YFR for the fleet is *not* equally distributed among the aircraft.

35.   A statistical analysis has been conducted in order to determine how the flying hours are distributed among the aircraft. Data used for the analysis was the data corresponding to the year 1999. More precisely, 1464 entries have been considered, consisting of the number of hours flown monthly by each of the aircraft in the fleet (122 A/C × 12 months). "BestFit" (see Ref. 4), the software used to perform the analysis, showed that no distribution was a good enough fit for our data (all distributions were rejected). Therefore, since the assumption that the monthly flying hours follows a specific distribution could not be made, other ways to implement the randomness of the process had to be found. The solution adopted was to implement a somewhat "controlled" randomness, so that the numbers produced by the simulation program be as reflective as possible of the real life process. The rules for "control" implemented in the model version documented here are based on the reality of the year 1999. However, given that in 1999 special deployments occurred (such as the ones to Aviano, Italy), a separate analysis was conducted for the year 1998. The results fully support the modeling technique that has been chosen to implement the process of assigning flying

- 15 -

hours to aircraft, as it will be seen in the following paragraph. However, should operations dramatically change in the future (not in the sense of YFR, but in the sense of how the YFR is distributed among the aircraft), new control rules would have to be implemented. A periodical revision of these rules is recommended.

36.   Data corresponding to the year 1999 showed that 93.5 % of aircraft that flew during this year, were flying between 1 and 43 hours, while the mean of the monthly hours flown was 21.74 hours for the whole of 1999. The remaining aircraft (6.5 %) flew between 44 and 120 hours. Analysis conducted on data corresponding to the year 1998 showed that 93.2 % of aircraft that flew during this year, were flying between 1 and 40 hours, while the mean number of monthly flying hours was 20.21 for the whole of 1998. The remaining aircraft (6.8 %) flew between 40 and 56 hours. As it can be seen, in both years, the vast majority of aircraft flew between 1 and twice the mean for the year. The difference between year 1998 and year 1999 is that in 1998 no aircraft flew more than 56 hours, while in 1999 some aircraft flew up to 120 hours. This is explained by the fact that, in 1999, the aircraft deployed to Aviano flew longer missions.

37.   In the simulation, the 93.5 % model was chosen to apply to *all* aircraft in the fleet. This means that the model will never assign to an aircraft a high number of flying hours (although such assignments did occur in approximately 6 - 7 % of the aircraft in the sample data). However, this does not represent a limitation of the model, because of the large number of iterations that are performed (at least 500). For the sake of the argument, let us assume that the model would be capable of assigning high numbers of flying hours to some of the aircraft. Suppose that, at the first iteration, aircraft j was one of the few aircraft that was assigned such a high number of flying hours. To aircraft j, at each iteration, a different number of flying hours is assigned. The reason several iterations are performed is to be able to determine the *most probable* outcome, from a number of *possible* outcomes, which is done by calculating the statistical mean. Therefore, given that what really counts is the average, it would take a good many number of iterations in which high numbers of flying hours would be assigned to aircraft j, in order to make a difference in the average (for this aircraft). This, of course, is very unlikely to happen

- 16 -

because of the randomness of the process of assigning flying hours to aircraft (aircraft j cannot possibly be selected over and over again as an aircraft to be assigned significant amounts of flying hours). For this reason, it was considered that applying the 93.5 % model to *all* aircraft in the fleet was fully justified.

38.   As a result of these observations, it was decided to assign to aircraft tail numbers, flying hours that were generated as random numbers belonging to the interval $[1, \alpha]$, where α is equal to twice the mean number of flying hours per aircraft for the current month, i.e.,

$$\alpha = 2 \times \frac{YFR/12}{\text{number of aircraft that are flying in the current month}} \qquad (5)$$

In other words, it was considered that the number of hours flown by the fleet is approximately the same each month, calculated such that the total for the year equals the YFR. At the face of it, the assumption that the number of flying hours for the fleet is approximately the same in each month is a very questionable one. However, the reader is reminded that the model was conceived to simulate the evolution of the fleet *year by year* (rather than month my month). The importance of this observation and its bearing upon the "questionable" assumption will become clearer after reading the next two paragraphs.

39.   In the previous versions of the program, all key functions of the model had a horizon of one year. Flying hours were allocated to the aircraft on a yearly basis. Fatigue was also allocated yearly. Crash attrition was equally based on a one year span, as were the maintenance activities. More precisely, on an annual basis, maintenance activities were modeled by assigning a certain number of months to be spent in the plant to some of the aircraft in the fleet. This is a random process based on historical observations. Modeling the maintenance activities was important mainly in order to capture the fact that, while in plant, these aircraft did not fly and therefore did not accumulate as much fatigue as aircraft that were operational the whole year. In this respect, it was sufficient to know that an aircraft was in plant for x months during the year, so that the fatigue accumulated could be adjusted accordingly.

- 17 -

40.   Currently, the modeling philosophy of the maintenance activities has changed, due to two main reasons: (1) more knowledge became available on the exact package of structural modifications (and implicitly its duration) that each aircraft is supposed to undergo, and (2) a requirement for better tracking of aircraft in 2nd and 3rd line maintenance. For these two reasons, the present implementation of the model has been changed, to allow a more precise simulation of 2nd and 3rd line maintenance activities. As an example, with regards to 3rd line maintenance, now, instead of knowing that aircraft j is in plant for 4 months during a particular year, we know that aircraft j is in plant from June through September inclusively. In addition, we also have the capability to model a situation in which aircraft j would be in the plant from October the current year, to January of the next year. However, in order to make this gain in precision possible, it was necessary to change the process of the allocation of flying hours from yearly to monthly. In other words, the assignment of flying hours on a monthly basis was done in order to permit a better handling of the maintenance activities. In the program, only the maintenance related functions make use of this monthly allocation of flying hours. For any other function, only the yearly total number of flying hours counts. For these reasons, the assumption that the total number of flying hours for the fleet is approximately the same for each month is valid only when applied to the maintenance related functions. Therefore, the imprecision introduced by this assumption is transferred solely to the functions regarding the 2nd and 3rd line maintenance activities and has no effect on any other results obtained from the model.

41.   As a final note on the subject of flying hours, it was previously stated that random numbers were assigned such as the monthly flying hours for the whole fleet (and implicitly the YFR) is reached within ± 3 %. This is checked within the function "*hours_flown*": in case the first set of random numbers does not satisfy the ± 3 % condition, a new set is obtained (by means of a new iteration) until the condition is satisfied.

- 18 -

AUXILIARY FUNCTIONS

42.   Six functions are grouped under this category. Five of them will be described in this section, in the order they appear in the program. The sixth one, implementing the normal distribution, will be described in Annex E, since its relevance to the functioning of the model is relatively minor, compared to the other functions.

The function "*read_database*"

43.   The program uses two data files as input for the model (see Annex A). These data files have to be periodically updated (monthly, or whenever a new simulation is done), and they should also be correlated, which will be explained in the next paragraph. The first file contains data regarding the number of hours flown and the FLEI accumulated by each aircraft in the fleet. The most recent values available should be used. The program reads this file and initializes the corresponding parameters of the simulation accordingly. The second file contains data regarding the number of flying hours on each airframe at the last periodical inspection.

44.   The two files should be correlated because they both set the initial conditions for the simulation, and *all* parameters of the model have to be initialized to represent the same date.

The function "present_status_LEP"

45.   In this version of the model (valid as of March 2001), this function indicates which aircraft have already received the first package of structural modifications (CP1). In the future, complementary information should be added, in order to reflect the most current situation concerning the Life Extension Program (LEP). Ideally, this function should be updated each time a new scenario is implemented.

46.   The analyst should be aware of the fact that, failure to update this function will not stop the model from functioning, but it may compromise, to a lesser or greater extent,

- 19 -

the results. The idea is that, by updating this function, the model will be fed with the most recent *reality* concerning the LEP. If not updated, the model would be fed with past reality and would have to make up for the missing data by using its own *forecasts* for the period from the last time the function had been updated to the present time. Therefore, every effort should be made to update this function regularly, thus avoiding the use of forecasted data when real data is available.

The function "*set_YFR*"

47.   As its name implies, this function establishes the YFR to be used each year (over the time horizon of the simulation), as indicated by the user. The function, as it is implemented, allows the YFR to vary in three levels, over three distinct periods. However, this may be easily changed, depending on the scenario to implement.

The function "*time_passes*"

48.   This function is necessary for the "housekeeping" of the program. More precisely, this function simulates the passage of time for the aircraft situated in 2nd or 3rd line maintenance.

The function "statistic_results"

49.   This function calculates mean values of some of the parameters of interest in the simulation and records them in output files bearing self-descriptive names. It should be noted that the numbers recorded in the output files actually represent the mean values rounded to the nearest integer, since one cannot speak of fractions of an aircraft. This will explain why, for example, the total number of aircraft remaining in a certain year may not be equal to the sum of the number of singles and the number of duals remaining in that year. A numerical example will clearly show how this may occur and why it should not be a reason for concern regarding the correctness of the program. Under a certain scenario, the mean value of the total number of aircraft remaining in the year 2012 was 70.51. In the corresponding output file, this value has been rounded to the nearest

- 20 -

integer, which is 71. The mean number of singles remaining in the same year was 53.06, rounded to 53, while the mean number of duals remaining in that year was 17.45, rounded in the output file to 17. Therefore, although 53.06 + 17.45 = 70.51, showing that the program works properly, the sum of the rounded numbers 53 + 17 = 70.

50. It should also be noted that each new simulation of a scenario overwrites the output files. If several scenarios are to be run consecutively, after each execution of the model the results should be copied to a different file.

51. The parameters considered of interest in the present version of the model are the following:

    a) the total number of aircraft remaining operational each year;

    b) the number of single / dual aircraft remaining operational each year;

    c) the cumulative number of aircraft that have crashed;

    d) the cumulative number of aircraft that have fatigued-out;

    e) the number of aircraft that are in second line maintenance, on a monthly basis;

    f) the number of aircraft that are in third line maintenance, on a monthly basis; and

    g) the number of aircraft waiting to be accepted in third line maintenance, on a monthly basis.

Other parameters may easily be added, along with the appropriate calculations, depending on the user's needs.

52. One final note concerning the output files of the function "*statistic_results*" is that the analyst should periodically modify the base year in this function, in order to obtain meaningful dates in the output files. It is important to know that, in the program, "*year = 0*" represents the first year following the present date. The results corresponding to "*year = 0*" describe the situation *after* the first year of operation of the fleet. For example, if an analysis is conducted in March 2001, "*year = 0*" represents the period March 2001 – March 2002, while the corresponding result that is recorded in the output file describes (forecasts) the situation as of March 2002. To be able to properly record these years in

- 21 -

the output files, one has to adjust the base year accordingly (i.e., using the formula *base year = present year + 1*).

- 22 -

PART II: PRACTICAL USE OF THE MODEL

53. This section is dedicated to the practical use of the model and is organized in three parts. The first part contains a description of the scenario that is presently considered as the most probable "way ahead" for the CF-18 fleet and, and which is called, for this reason, the *baseline scenario*. The second part provides a brief summary of the core capabilities of the model. Finally, the third part contains a set of more elaborate practical results that have been obtained through the model and that were considered of special interest at the moment this study was done. They constitute examples of how the model can be used, and they are by no means exhaustive. The model is extremely versatile, and, once its core components are understood, virtually any scenario can be implemented, provided that the analyst is capable of making the necessary modifications/adjustments to the model.

BASELINE SCENARIO

54. The baseline scenario is based on the following assumptions:

a) YFR = 18300 hours for FY 2000/2001; YFR = 17800 hours for FY 2001/2002; YFR = 16000 hours for the following years;

b) From the 122 aircraft presently in the fleet, 80 aircraft have been selected for upkeep and modernization. Four of these (747, 749, 758 and 759) are waiting for a centre barrel replacement (CBR), while the other 76 aircraft (728, 729, 730, 731, 732, 733, 734, 735, 736, 738, 739, 740, 741, 742, 743, 744, 745, 746, 748, 750, 751, 752, 753, 754, 756, 757, 760, 761, 762, 763, 766, 767, 769, 770, 771, 774, 775, 776, 777, 778, 780, 781, 782, 783, 784, 785, 786, 787, 788, 789, 790, 791, 793, 794, 795, 796, 797, 798, 917, 918, 923, 924, 925, 926, 927, 928, 930, 932, 933, 934, 935, 936, 937, 938, 939, 940) are currently receiving or will be receiving a specific set of structural modifications;

- 23 -

c) The remaining 42 aircraft currently in the fleet that have not been selected for upkeep and modernization have already been retired or will be retired according to a retirement scheme based on various FLEI thresholds. Specifically, aircraft having tail numbers 722 and 764 are already retired. The early production dual aircraft 901, 902, 903, 904 and 905 are to be retired when they reach the FLEI threshold of 0.52. A retirement threshold of 0.56 has been set for aircraft 701, 702, 703, 705, 706, 707, 710, 711, 712, 718, 719, 723, 724, 906, 907, 908, 909, 910, 911, 912, 913, 914, 915, 916, 920, 921, 922, 929 and 931. Another four aircraft, 709, 716, 720 and 725 will be retired upon reaching a threshold of 0.645, while two other aircraft, 708 and 727 will be retired upon reaching a threshold of 0.733;

d) Fatigue consumption rates: 0.135 ± 0.06 FLEI/1000 hours (normal distribution);

e) Crash attrition is based on the statistical (Learning Curve) model;

f) Second line maintenance: All aircraft are eligible to go into second line maintenance for as long as they fly. An aircraft is due to second line maintenance at every 400 flying hours and these activities take one month;

g) Third line maintenance: Only aircraft selected for upkeep and modernization are eligible to the third line maintenance activities. Three packages of structural modifications have been defined (CP1, CP2 and CP3), but they will have to be implemented in *at most* two passes of each aircraft through the plant. The following combinations have been agreed upon between DND and Bombardier and these are the combinations considered in the baseline scenario:

    i. <u>Combination 1</u>: two passes

        1st pass = CP1; 2nd pass = CP2 + CP3 (combined);

    ii. <u>Combination 2</u>: two passes

        1st pass = CP1 + CP2 (combined); 2nd pass = CP3;

    iii. <u>Combination 3</u>: one pass

        CP1 + CP2 + CP3 (combined);

The first package of modifications (CP1) will be able to extend the fatigue life of the aircraft receiving it up to an FLEI of 0.645. Its duration is estimated to be seven months. The second package (CP2) will permit an extension of the fatigue

- 24 -

life of an aircraft up to a FLEI of 0.733 and its duration is estimated to be approximately seven months. The third package (CP3) will further extend the fatigue life up to a FLEI of 1.0 and it is estimated to take approximately five months. The modifications included in CP1 have already been completed from an engineering point of view. The modifications forming the second package (CP2) are expected to be finished by 2002, while those included in the third package are estimated to be ready by 2003 – 2004. It has been decided between DND and Bombardier that it would not make economic sense that aircraft that already have a FLEI beyond 0.57 be among the aircraft receiving the "combination 1". Therefore, in the baseline scenario, these aircraft are modeled as not eligible for "combination 1", but rather for "combination 2". In the meantime, until CP2 is completed, these aircraft are restricted from flying. It has been considered, based on the induction plan for the period September 2000 – January 2003 provided by Bombardier, that the average number of aircraft that Bombardier has the capacity to work on at the same time is 12. In the model, if an aircraft is due to be sent to the plant in a given month, it will go if the number of aircraft already in the plant is less than 12. If not, the aircraft will have to wait (while being restricted from flying) until a place becomes free at Bombardier. The assumptions detailed here are valid for all aircraft but four: 747, 749, 758 and 759. These aircraft are awaiting a centre barrel replacement. At the time the model was last updated, no information was available concerning the availability and duration of CBR. In the model, similar thresholds and duration had been used, as for the rest of aircraft. In the baseline scenario, the structural modifications program is implemented in the most economical way in terms of dollars. More precisely, since in the first year only CP1 is available, all aircraft that are due for structural modifications *and* eligible for "combination 1" will receive "combination 1". Once CP2 becomes available, "combination 1" will become obsolete and any aircraft that are due and eligible for modifications will receive "combination 2". Finally, once CP3 becomes available, "combination 2" becomes obsolete as well, and all aircraft that have not begun any structural modifications program will receive "combination

- 25 -

3" (even if they were already assigned a different program at inception), since this is the cheapest option: and

h) The Incremental Modernization Program (IMP) was not modeled, since very little information was available about this program at the time the model was last updated

BRIEF SUMMARY OF THE CORE CAPABILITIES OF THE MODEL

55   This brief summary of the core capabilities of the model concentrates mostly on the features that were recently added to the model and are not documented elsewhere. For a full comprehension of *all* the model's capabilities, the analyst should first understand the "philosophy" of the model and the functioning of each of its modules/functions, and review some of the past analyses and results than can be found in Ref 1, 2 and 3.

56.   Hereafter, an illustration of the types of output data presently provided by the model is given, along with a brief discussion regarding the handling of the most important parameter of the model, the YFR

57.   Figures 1 and 2 illustrate some of the output data that is provided by the model. Figure 1 displays the number of aircraft "remaining" in any year. There are three curves on this graph, corresponding to single A/C, dual A/C, and "all" A/C. Figure 2 shows the cumulative number of lost aircraft. The two curves on this graph plot the number aircraft that were lost due to either crash attrition or fatigue attrition.

- 26 -

No. of Remaining Aircraft by Type
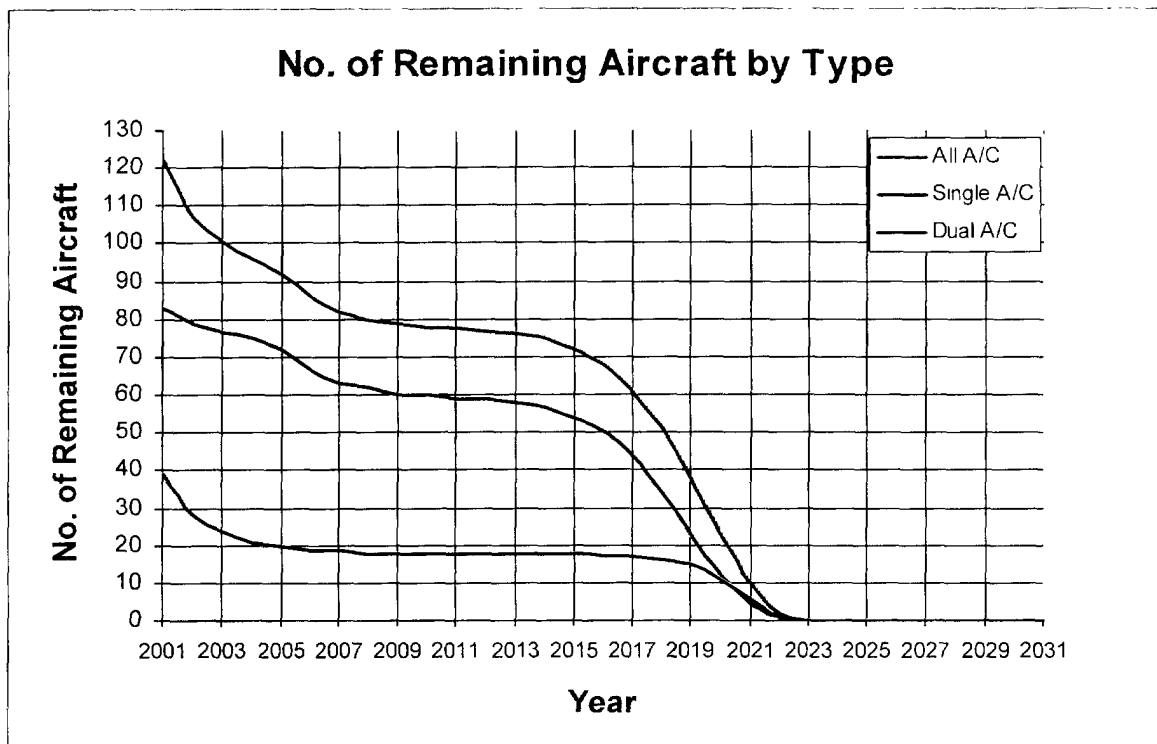
Figure 1. Number of remaining aircraft by type

Cumulative Number of Lost Aircraft Due to Fatigue or Crash Attrition
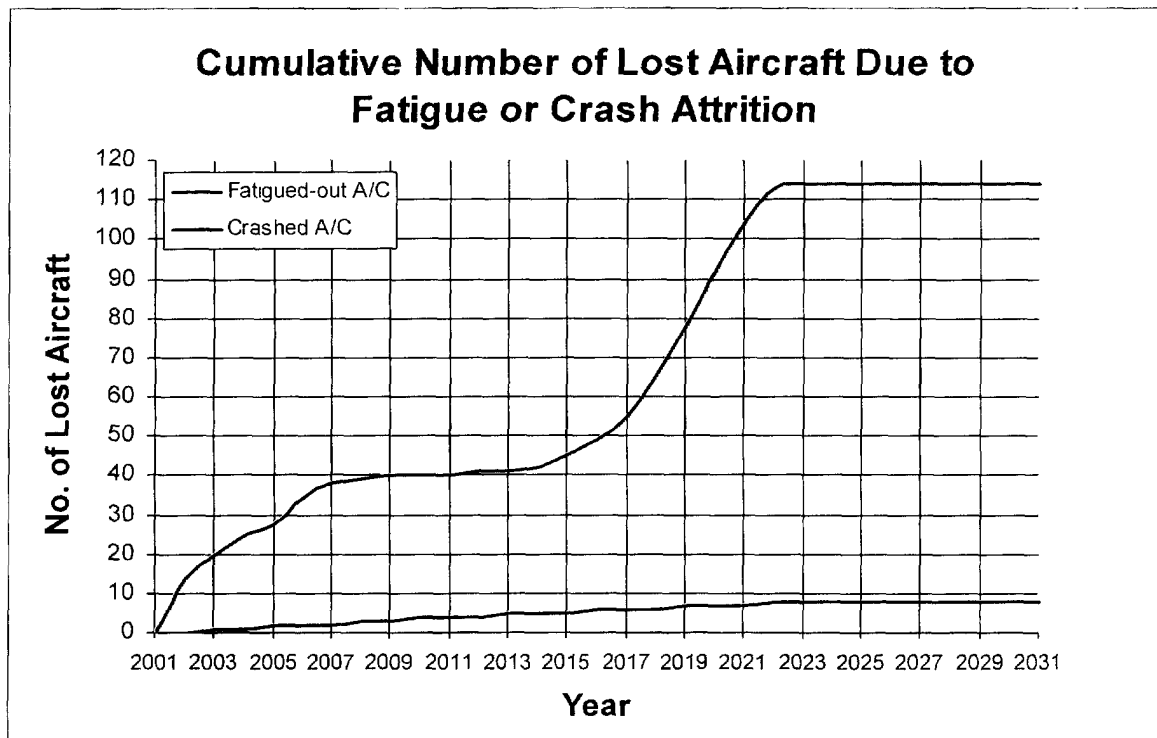
Figure 2. Cumulative number of aircraft losses

- 27 -

58.   Sample output data that is recorded on a monthly basis is presented in Tables I and II.

TABLE I
NUMBER OF OPERATIONAL AIRCRAFT
(BASELINE SCENARIO)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2001	86	97	97	97	96	94	93	104	103	102	101	100
2002	86	86	91	90	90	90	89	89	88	94	93	92
2003	85	84	83	83	83	83	84	84	84	83	83	85
2004	80	79	78	77	76	75	74	75	75	74	73	73
2005	69	68	67	66	65	64	65	66	65	64	63	62
2006	56	54	54	53	52	51	51	52	52	52	51	51
2007	47	46	47	46	46	46	47	49	49	50	50	51
2008	49	50	51	51	52	52	53	55	55	56	56	56
2009	56	56	57	58	58	59	59	60	61	61	62	62
2010	62	63	65	65	66	67	67	69	69	70	70	70
2011	70	70	71	71	72	72	72	72	73	73	73	73
2012	73	73	73	73	73	73	73	73	73	73	73	73
2013	72	72	72	72	72	73	73	73	73	73	73	73
2014	71	71	71	71	71	71	71	71	71	71	71	71
2015	69	68	68	68	68	68	69	69	69	68	69	69
2016	64	64	64	64	64	64	64	64	64	64	64	64
2017	57	57	57	57	57	57	57	57	57	57	57	57
2018	48	47	47	47	47	47	47	47	47	47	47	47
2019	35	34	34	34	34	34	34	34	34	34	34	34
2020	21	20	19	19	19	20	20	20	20	20	20	20
2021	9	7	7	8	8	8	8	8	8	8	8	8
2022	1	1	1	1	1	1	1	1	1	1	1	1
2023	0	0	0	0	0	0	0	0	0	0	0	0
...	0	0	0	0	0	0	0	0	0	0	0	0
2030	0	0	0	0	0	0	0	0	0	0	0	0

59.   Table 1 shows the expected number of operational aircraft each month, for the next 30 years. The reader is reminded that an aircraft is called "operational" when it is capable of actually flying missions (i.e., the aircraft is not crashed, not fatigued-out, not in second or third line maintenance and not waiting to go into third line maintenance). This type of table can be very useful for the planning of training activities. Besides, it may also act as an alarm bell, in case some of the levels of operational aircraft attained are not acceptable.

- 28 -

60.   Table 2 shows the number of aircraft that are waiting to go into third line maintenance (i.e., aircraft that are restricted from flying because they are due to receive some package of structural modifications, but cannot go into the plant either, because the plant is already working at capacity). This table shows a real problem beginning in 2005 and continuing until 2008 inclusive. It also explains the low levels of operational aircraft during the same period.

TABLE II
NUMBER OF AIRCRAFT WAITING TO GO INTO 3RD LINE MAINTENANCE
(BASELINE SCENARIO)

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2001	7	8	9	10	12	13	14	5	6	6	7	7
2002	7	7	4	4	4	4	5	5	6	3	3	3
2003	1	1	1	2	2	2	2	2	3	3	4	2
2004	2	2	3	4	4	5	6	5	5	6	7	7
2005	8	8	9	10	11	12	12	11	12	12	14	14
2006	15	16	16	17	18	20	19	18	19	19	20	20
2007	20	20	20	20	20	21	19	18	17	17	17	16
2008	16	15	14	14	13	13	12	10	9	9	9	8
2009	7	7	6	5	5	5	4	3	3	2	2	2
2010	1	1	0	0	0	0	0	0	0	0	0	0
2011	0	0	0	0	0	0	0	0	0	0	0	0
...	0	0	0	0	0	0	0	0	0	0	0	0
2030	0	0	0	0	0	0	0	0	0	0	0	0

61.   YFR is one of the most important factors affecting the structural life of the CF-18 fleet. Figure 3 displays three scenarios: the baseline scenario (as described earlier) and two other scenarios that are identical to the baseline scenario, but for one parameter, the YFR. More precisely, in Scenario 1, YFR is set at 16000 hours, from the first to the last year of the simulation. In Scenario 2, YFR is set at 18300 hours, again, from the first to the last year of the simulation. Just as a reminder, the baseline scenario is characterized by an YFR of 18300 hours for the first year, 17800 hours for the second year and 16000 hours for the following years[5]. Although the difference between these three scenarios is

---

[5] In a previous version of the model, YFR was automatically decreased when the number of remaining aircraft fell beyond a certain threshold. However, in the present version of CF-18 SLAM, YFR is kept constant regardless of the number of remaining aircraft, as required by the sponsor (see Ref.8)

- 29 -

quite slight, the difference between the number of aircraft remaining in each scenario is noticeable. From a programming point of view, the handling of YFR is described in Annex C (along with various other parameters).

Impact of YFR on the Number of Remaining Aircraft
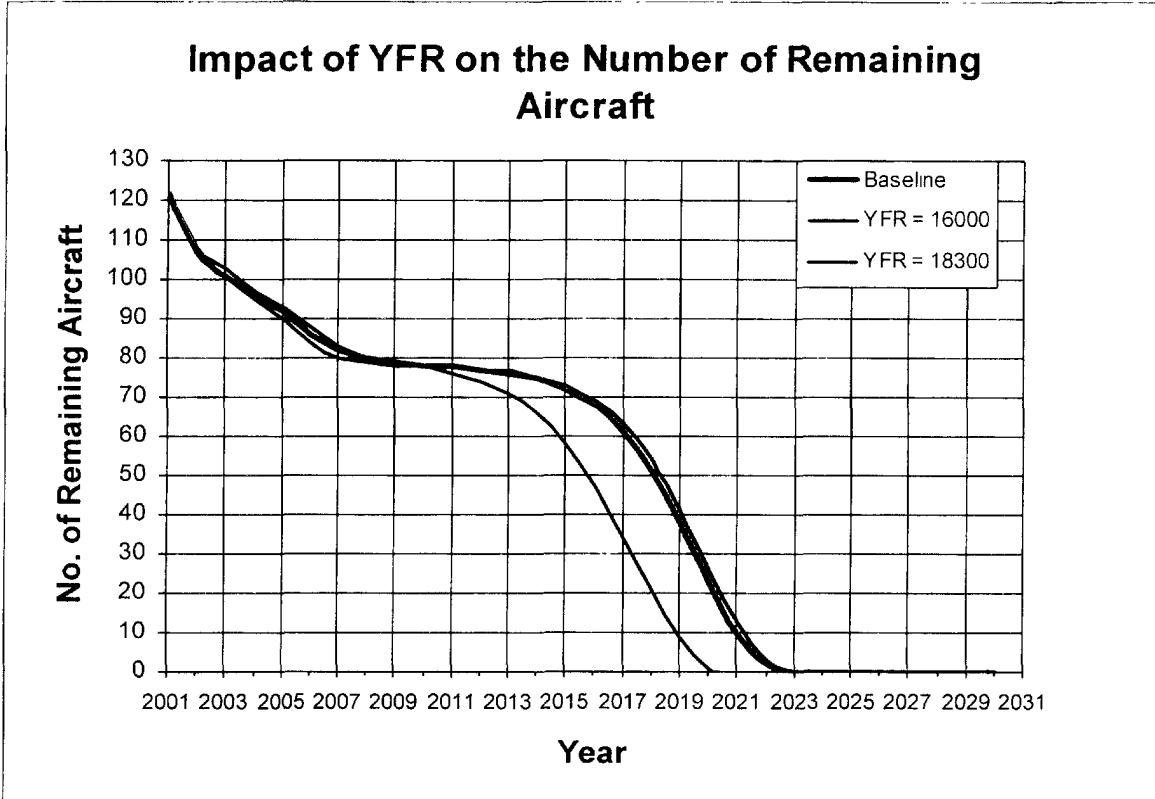
Figure 3. Impact of YFR on the number of remaining aircraft

62. Historically, most of the "what if" analyses that were requested by the CF-18 community concerned studying the impact that different YFR patterns may have on the life of the CF-18 fleet. Another set of analyses concerned the impact of IFOSTP on the life of the fleet. A smaller number of past analyses were directed towards determining the impact on the fleet of having different fatigue rates, or operating fleet reductions at various points in time, or upgrading different sets of aircraft in different ways. The model is capable of simulating all these situations and more. Recently, new features were added to the model that permit handling of new factors such as the recent program of upkeep and modernization of the aircraft (see Annex C).

- 30 -

LATEST RESULTS OBTAINED THROUGH THE MODEL

Discussion on the number of remaining aircraft vs. the number of operational aircraft in the baseline scenario

63.   The number of remaining aircraft (i.e.. aircraft that did not crash and did not fatigue-out) has been the main measure used in the forecasting of the structural life of the CF-18 fleet. However, the use of this measure implicitly assumes that all the remaining aircraft are available aircraft that one can count on for training or operational missions But, with the Life Extension Program that the fleet is presently undergoing, this implicit assumption cannot be satisfied anymore. since, for long periods of time. a significant number of aircraft will be in the plant undergoing structural modifications and therefore not available for flying.   Based on this observation. another parameter has been introduced in the model: the number of *operational* aircraft. The "operational" aircraft are those aircraft that are available for flying missions. Formally, this measure is defined as the number of aircraft that did not crash or fatigue-out. that are not in second or third line maintenance and that are not restricted from flying for any other reason. This capability of the model for forecasting the number of operational aircraft can be very useful for planning purposes, mainly concerning the feasible YFR and the training capacity that is expected to be available in any particular year.

64.   CF-18 SLAM determines the number of remaining aircraft on a yearly basis. The number of operational aircraft is calculated on a monthly basis. In Figure 4. both parameters are plotted on the same graph. to give the reader an idea of what the difference between the number of remaining aircraft and the number of operational aircraft may be. In order to make the comparison easier on the graph. both parameters were calculated on a yearly basis. The yearly value of the number of operational aircraft is calculated as the average value over the 12 months of the year.

65.   From Figure 4 it can be seen that not only is the number of operational aircraft smaller than the number of remaining aircraft. which was to be expected. but the two

- 31 -

curves do not have the same profile at all, which is a concern. Or, at least, the latter result could not have been "guessed" (or expected), without a formal analysis.

66.   Given the present FLEI distribution, the present estimations of the CP1, CP2 and CP3 duration, and the present level of effort that Bombardier is able to provide, one may note that, by 2007, although it is expected to have more than 80 aircraft remaining, less than 50 of them are expected to actually be "operational". The rest will be either in plant, undergoing structural modifications, or waiting to go in plant (and in the meantime restricted from flying) This may represent an important problem, in the sense that the training quality may be compromised by lack of aircraft. By 2011, when the Life Extension Program will basically be finished (again, according to the assumptions used in the baseline scenario), the two curves begin to have the same profile. The number of operational aircraft will be only very slightly under the number of remaining aircraft, the difference being due to aircraft in second line maintenance.
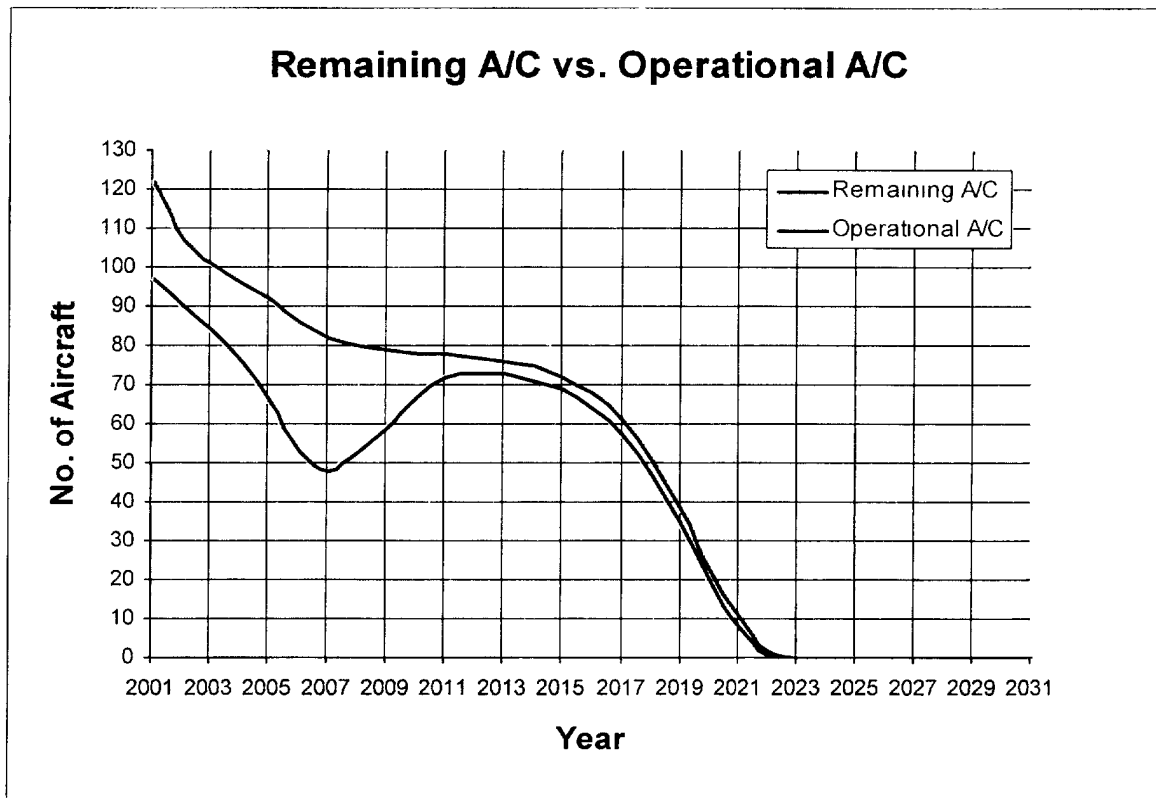
Figure 4. Number of Remaining Aircraft vs. Number of Operational Aircraft in the Baseline Scenario

- 32 -

Discussion on the uncertainties in the Life Extension Program (LEP) and in the Incremental Modernization Program (IMP)

67.   Since the engineering development of the structural modifications involved in the LEP is not yet finished, there is a certain degree of uncertainty in the assumptions used in the baseline scenario.   There is uncertainty the *duration* and also in the *year of availability* of two of the three packages of structural modifications, namely CP2 and CP3. In Figures 5 to 7, an assessment of these two types of uncertainties is provided.

68   Figure 5 plots three scenarios that were used to explore the potential effects that the uncertainty related to the duration of CP2 and CP3 may have.   The blue line represents the baseline scenario. The reader is reminded that in the baseline scenario it is considered that CP2 takes 7 months, while CP3 takes 5 months to be executed. The red line represents a more pessimistic scenario in which CP2 and CP3 are both one month longer than the baseline scenario (i.e., CP2 = 8 months, while CP3 = 6 months).
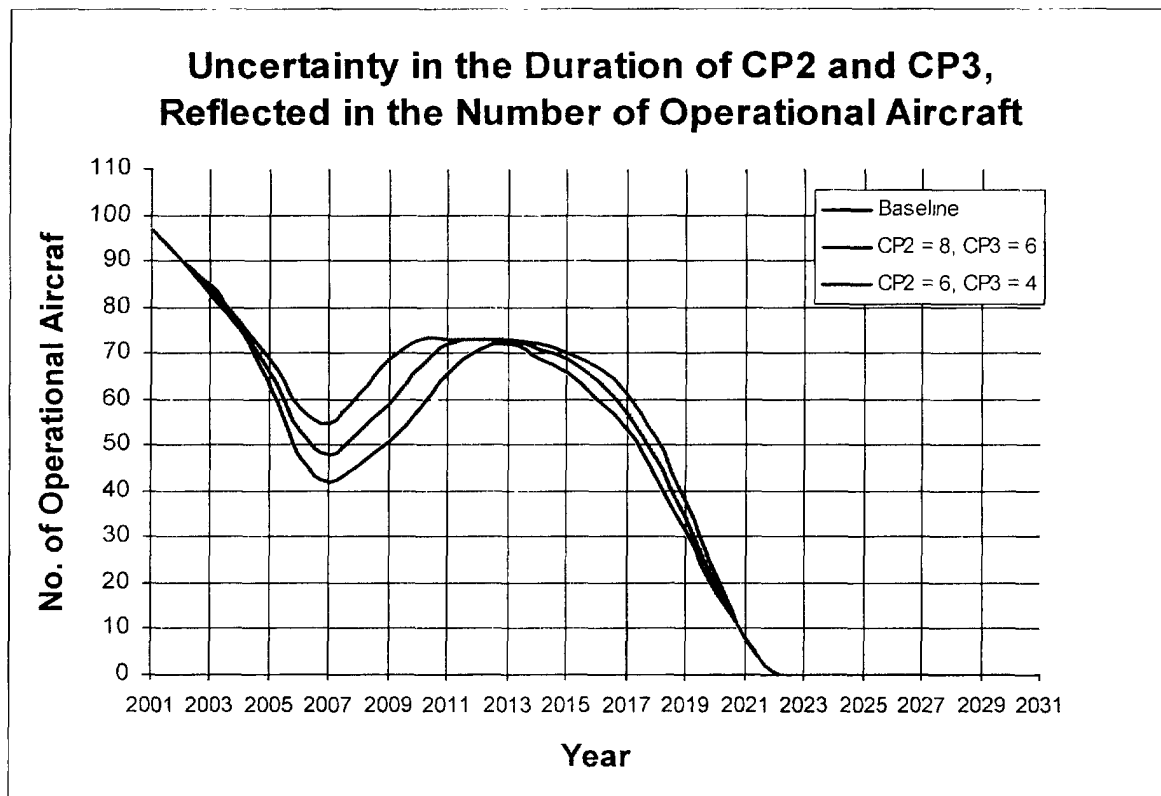
Figure 5. Assessing the impact that variations in the duration of CP2 and CP3 may have on the number of operational aircraft

- 33 -

Finally, the green line plots a more optimistic scenario, in which CP2 and CP3 are both one month shorter that the baseline scenario (i.e , CP2 = 6 months and CP3 = 4 months) It is to be noted that the differences between the baseline scenario and the two other scenarios are not significant and all scenarios should be viewed as having very good chances of actually occurring.

69.   Several observations can be made based on Figure 5. The first observation is that, although the differences between the three scenarios are not very important, the impact on the number of operational aircraft is quite noticeable. For example, in the more pessimistic scenario, for a period of approximately six years (2006 – 2011), the number of operational aircraft is approximately six aircraft less than in the baseline scenario, which may have a significant impact on the training capabilities offered to the CF-18 community. A second observation is that the implementation of the CP3 package will be finished approximately two years earlier in the baseline scenario than in the pessimistic scenario. The third observation is that, even after the implementation of CP3 is completed in all scenarios, there is still a slight difference in the number of operational aircraft, because it is a long term effect in that, for a long period of time, the difference between the numbers of operational aircraft in each scenario was significant. To clarify this idea, let us compare the more pessimistic scenario with the baseline scenario. For approximately six years, in the more pessimistic scenario there were six operational aircraft less than in the baseline scenario. This means that the YFR was "flown" by less aircraft in the pessimistic scenario and therefore each of the operational aircraft in this scenario flew more hours than the operational aircraft in the baseline scenario, with the direct result of accumulating more fatigue. Consequently, some aircraft will fatigue-out more rapidly in the more pessimistic scenario than in the baseline case.

70.   Although the most important impact that variations in the duration of CP2 and CP3 may have, is on the number of operational aircraft during the period in which CP2 and CP3 are implemented, an impact on the number of remaining aircraft also exists and this is displayed in Figure 6.

- 34 -

71   Figure 6 shows that variations in the duration of CP2 and CP3 do have an impact on the number of remaining aircraft. However, this impact is quite slight and manifests itself only *after* all third line maintenance activities consisting in structural modifications are completed.
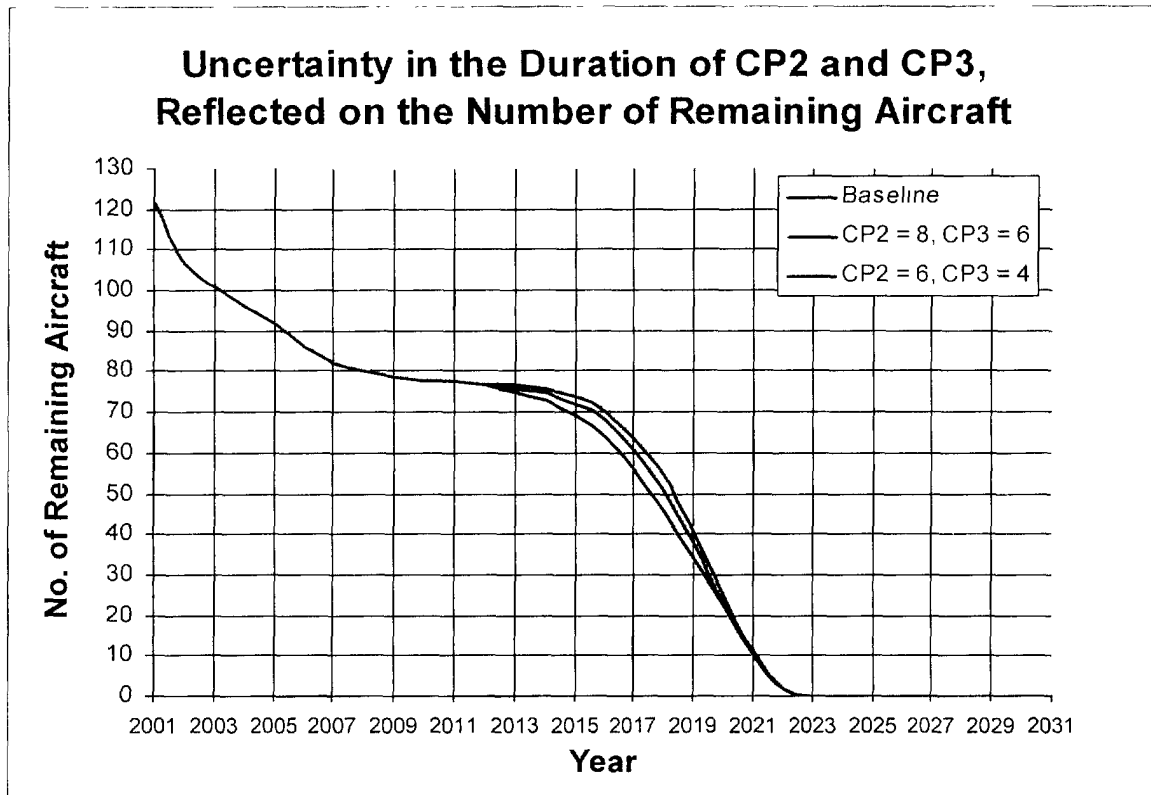
Figure 6. Assessing the impact that variations in the duration of CP2 and CP3 may have on the number of remaining aircraft

72.   As previously discussed, the structural modifications included in CP2 and CP3 are still under development. At the date of this report, it is estimated that CP2 will be ready to implement by 2002, while CP3 will become available in 2003. However, there is a certain degree of uncertainty related to these dates and therefore, an assessment was done on the implications of the deadlines not being met. Figure 7 shows some of the results. Virtually every combination of possible delays in the two deadlines can be implemented, but, in order to minimize the number of graphs and the number of curves included here, only the most probable scenarios are presented. It has been considered that, since the first deadline is relatively close to the date of this report, it has a good chance of being accurate. In the same line of reasoning, since the deadline for CP3 is more remote, it has

- 35 -

a good chance of not being accurate. Therefore, besides the baseline scenario, in Figure 7 two other scenarios are displayed: the first represents a situation in which CP3 would be delayed by one year, while the second represents a situation in which CP3 would be delayed by two years (compared to the baseline scenario).

Uncertainty in the Year of Availability of CP3, Reflected in the Number of Operational Aircraft
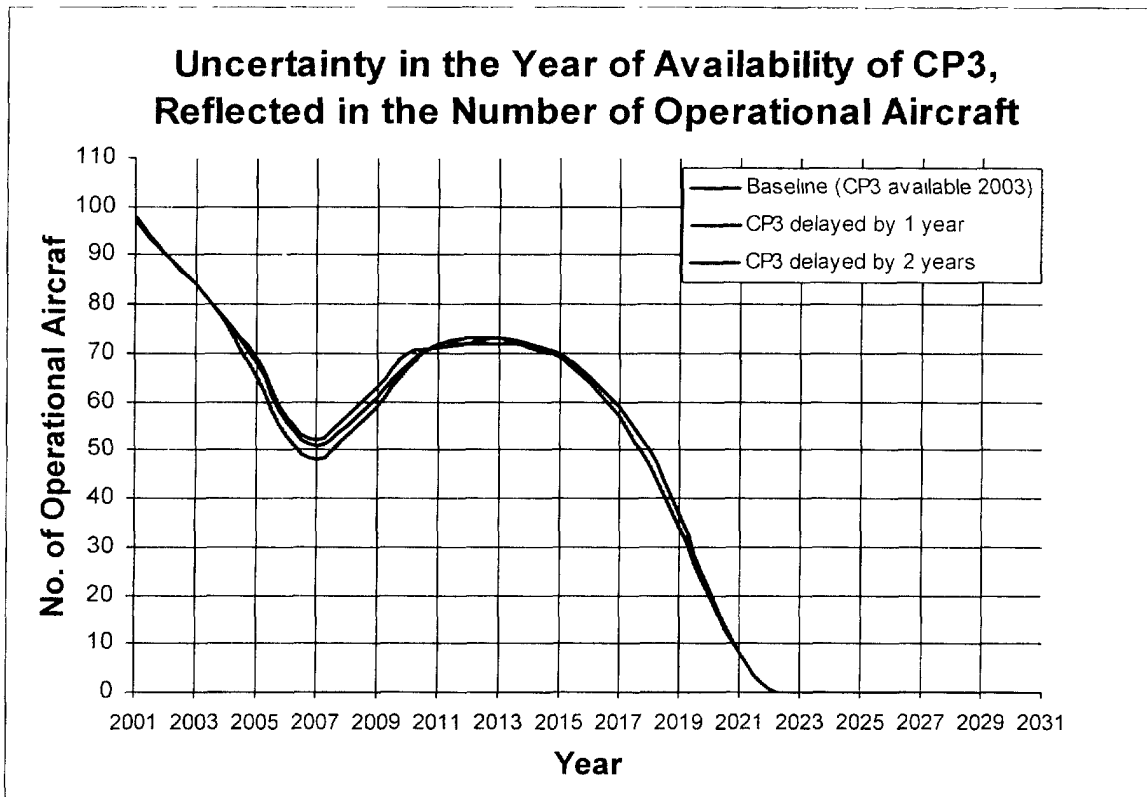
Figure 7. Assessing the impact that potential delays in the engineering development of CP3 may have on the number of operational aircraft

73. From Figure 7, it can be seen that a delay in the year CP3 becomes available may be considered as having a very slight positive effect on the number of operational aircraft. The downside to this is that this scenario is more expensive than the baseline scenario, because more aircraft are assigned to the maintenance programs requiring two passes through the plant rather than the program requiring only one pass (until CP3 becomes available, the one pass program cannot be assigned to any aircraft). As a last observation, the impact of the delay in the availability of CP3 on the number of remaining aircraft is practically not noticeable at all, and for this reason the corresponding graph, showing three superimposed curves, was not included here.

- 36 -

74.   The uncertainty surrounding the Life Extension Program is also proven by the scenario presently used at Bombardier for forecasting purposes. This scenario is different from the initial agreement between DND and Bombardier, showing that LEP is still in the tailoring stage.   Although most of the assumptions used in this scenario are identical to those described earlier in the baseline scenario, there is one assumption that is fundamentally different.   In the scenario used by Bombardier, every aircraft from the group of 80 selected for upkeep and modernization will first receive CP1, then fly for a while (usually, but not always, until a FLEI threshold of 0.645 is reached), and then return to the plant to receive the second and final package, called "CP1 Complement" CP1 Complement is equivalent to CP2 and CP3 combined, as they were defined in the baseline scenario.

75   The modeling approach adopted by Bombardier relies heavily on a deterministic intake scheduling scheme and accounts for all production constraints[6].   Given the differences between this approach and the probabilistic modeling approach described in this paper, it is to be expected that the results will never be the same. However, since important decisions are made based on such results, it is important to compare them and to be fully aware of the assumptions used in each case.

76   In Figure 8, the blue and green curves represent the number of operational aircraft corresponding to the scenario used by Bombardier, implemented using CF-18 SLAM and Bombardier's model respectively. The green curve stops in 2011 because that was all the data available from Bombardier. It should be noted that, due to the requirement of having a steady workload over the years, in Bombardier's simulation, some aircraft may receive CP1 or CP1 Complement before their due date, which does not happen in the CF-18 SLAM simulation. The red curve represents the number of operational aircraft under the baseline scenario with the CF-18 SLAM implementation.

---

[6] Examples of "production constraints" are the August and Christmas breaks observed at Bombardier, the requirement of having a steady workload over the years, etc

- 37 -

No. of Operational Aircraft under Two Different Scenarios

Legend:
— Bombardier Scenario / CF-18 SLAM Implementation
— Baseline Scenario / CF-18 SLAM Implementation
— Bombardier Scenario / Bombardier Implementation

Y-axis: No. of Operational Aircraft (0 to 100)
X-axis: Year (2001 to 2031)
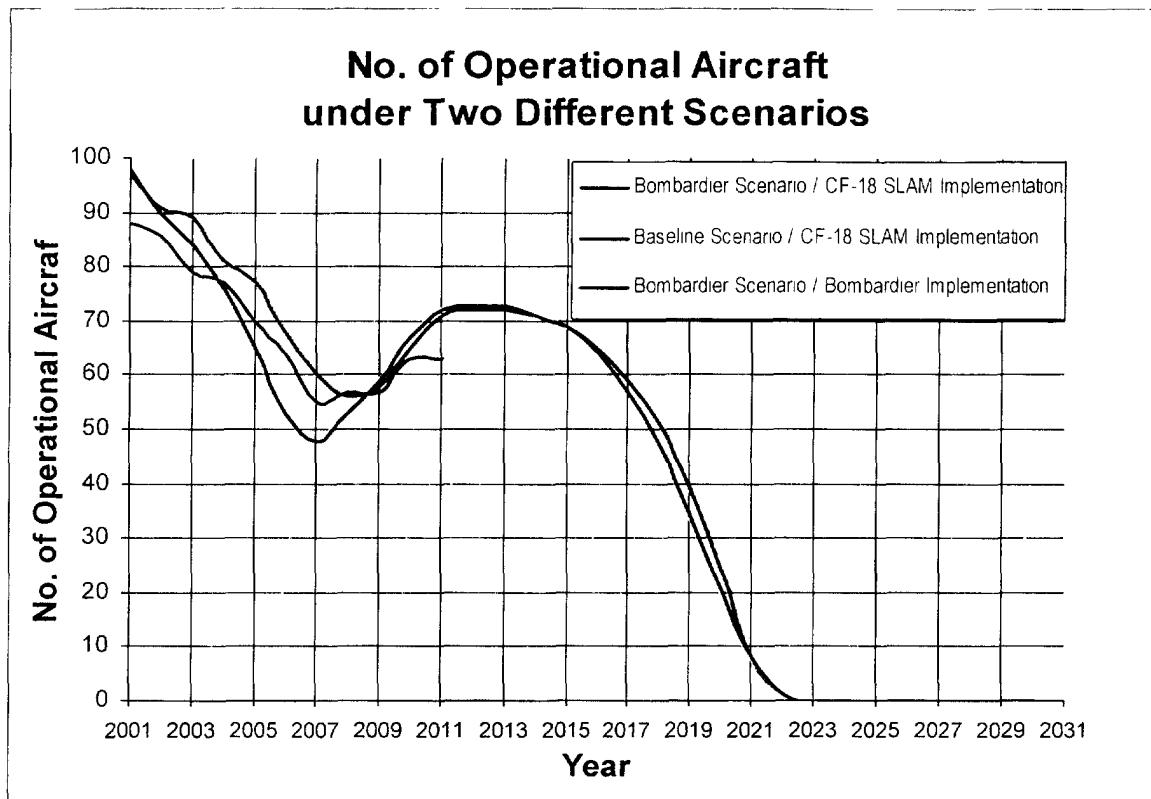
Figure 8. A comparison between DND and Bombardier results with respect to the number of operational aircraft

77.   Although a complete comparison of the two models is difficult to do, given the limited information that is available regarding Bombardier's model, it was considered of some value to attempt even a crude analysis. It is the author's opinion that the main strength of Bombardier's model is its capability to provide a very precise scheduling of the third line maintenance activities in the short to medium term. The model's main shortcoming is its relatively limited versatility, which is due to the spreadsheet approach that was adopted. Because the time required to implement a new scenario can be significant, the model would be difficult to use to test a large number of "what if" type scenarios, which could ultimately lead to insufficient analysis prior to making decisions. Bombardier's model weakness represents CF-18 SLAM's strength, while Bombardier's model strength represents CF-18 SLAM's weakness. More precisely, due to its much more flexible implementation in Visual C++, CF-18 SLAM is extremely versatile. Virtually any scenario can be implemented in a relatively short amount of time. On the other hand, CF-18 SLAM cannot offer, at least in its actual form, the degree of precision

- 38 -

that is required for planning and scheduling third line maintenance activities. It cannot be said that one model is bad while the other is good. since. although there is a significant overlap in their scope. there are also some differences. The Bombardier's model may be described as more of a planning or scheduling tool. heavily oriented towards the third line maintenance activities. while the model developed at DND may be described as more of a forecasting tool. with a horizon going beyond the third line maintenance activities Therefore. although it is true that the first model does offer to some degree a forecasting perspective. while the second is able to offer to some degree some planning information. the two models are largely complementary.

78   Very little is known about the Incremental Modernization Program. very little is known at writing (March 2001)   According to Ref. 7. this program. consisting in avionics upgrades that will make the CF-18 A/B attain the capabilities of the latest F/A–18 C/D models. will be implemented in two phases. The first phase is to commence by mid 2002 and to be completed by 2006. The second phase is scheduled to be completed by 2007-2008. It is expected that Boeing. the provider of the off-the-shelf modernization package customized for the CF-18. will subcontract the installation work to Bombardier Defence Services.   It is therefore expected that the avionics upgrades will be implemented on an aircraft while it is already in plant for structural modifications. although the two-phase approach may pose a problem in that sense.   With the data available at the present date (March 2001). it is difficult to produce anything different from a very rough estimation of the impact that IMP may have on the number of operational aircraft. The only thing that can be said right now is that there will be an impact. To get an idea of what this impact could be. the reader is referred to Figure 5. Assuming that IMP would take approximately two months per aircraft to implement. adding IMP to the structural modifications programs (as described in the baseline scenario) would be equivalent to the scenario represented by the red curve in Figure 5. where the durations of CP2 and CP3 were both increased by one month.

- 39 -

Discussion on the importance of the working capacity available at the plant

79.   The "working capacity" available at the plant is defined as the number of aircraft that can be worked upon at the same time. The model uses an average value of 12 aircraft for the baseline scenario, based on the induction plan provided by Bombardier for the period September 2000 -- January 2003. The maximum number of aircraft that could be worked upon at the same time depends basically on three elements: (1) the plant's capacity to physically accommodate a certain number of aircraft; (2) the human resources the plant is able to provide and (3) DND's budget. Apparently, the plant has enough space to accommodate a larger number of aircraft simultaneously. and since the two other elements are adjustable (at least in theory), an analysis was done to show what impact an increase in the working capacity at the plant may have on the number of operational aircraft.
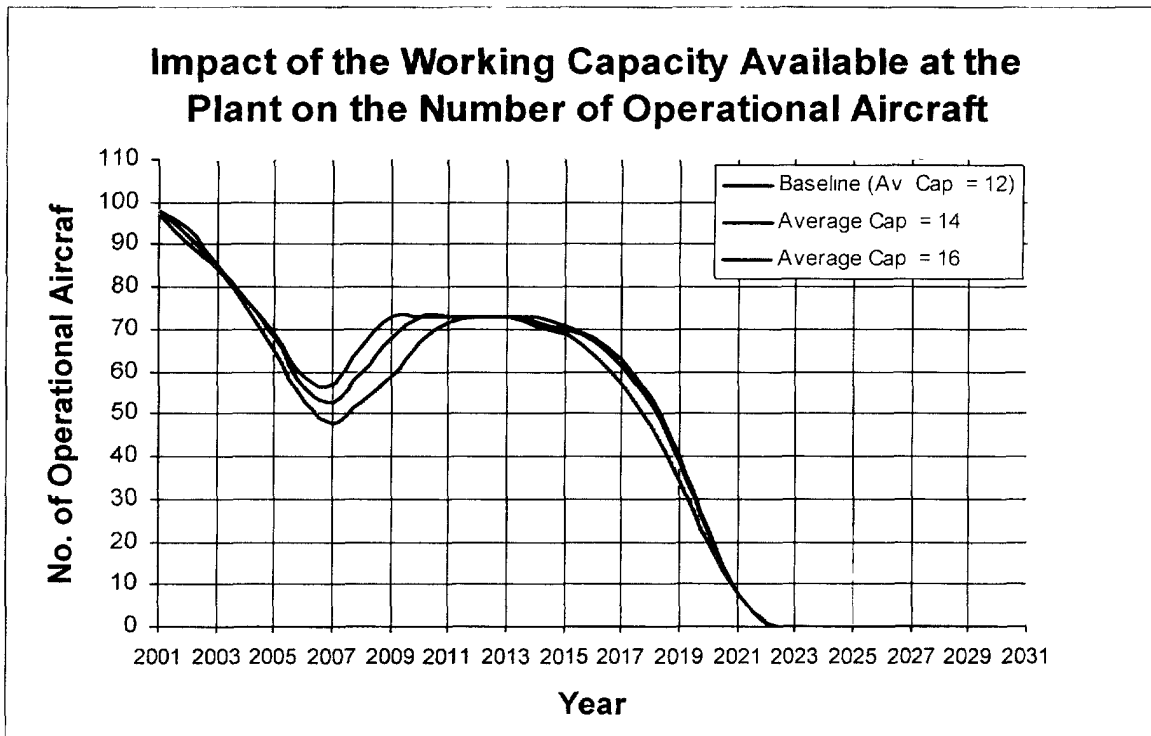
Figure 9. Assessing the impact that the working capacity available at the plant may have on the number of operational aircraft

- 40 -

Impact of the Working Capacity Available at the Plant on the Number of Remaining Aircraft
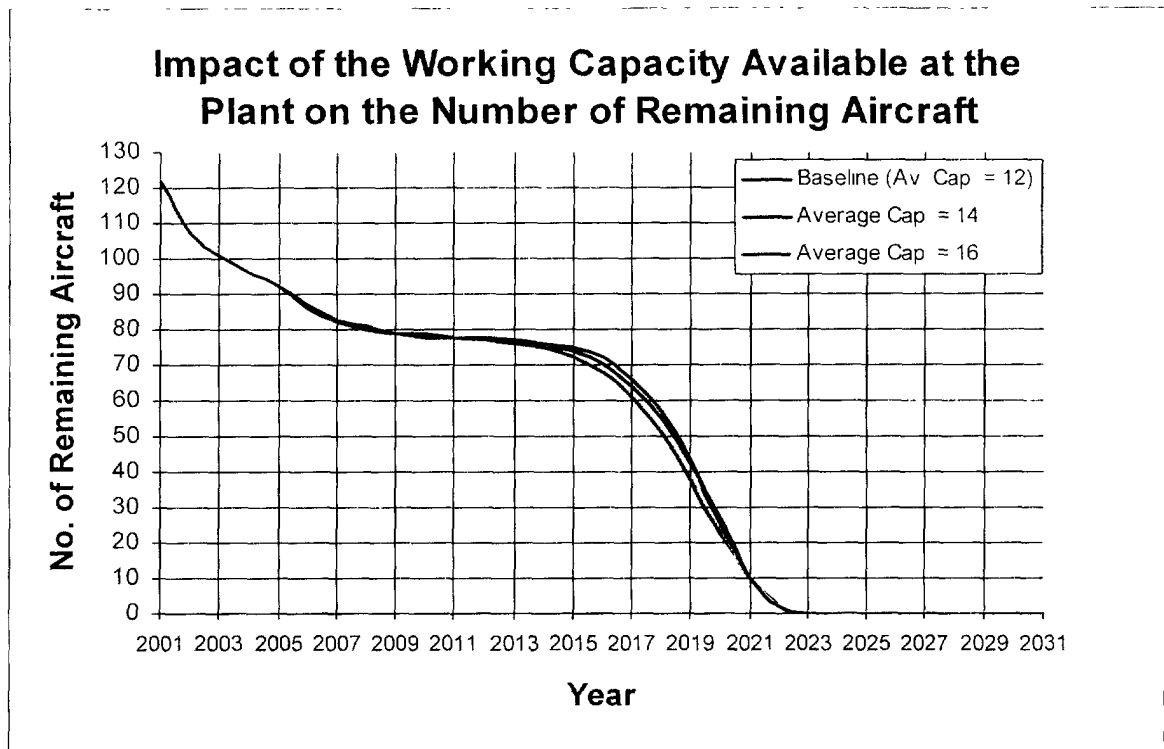
Figure 10. Assessing the impact that the working capacity available at the plant may have on the number of remaining aircraft

80.   Figure 9 displays three scenarios: the baseline scenario, characterized by an average working capacity of 12 aircraft, and two other scenarios characterized by an average working capacity of 14 and 16 aircraft, respectively. From the graph it can be seen that by increasing the working capacity at the plant, a significant improvement of the number of operational aircraft may be obtained. This improvement is quite important for a period of approximately five years (2006 - 2010), and slight but still noticeable in the following years

81.   Figure 10 shows that increasing the work capacity available at the plant offers only a very slight benefit in what concerns the number of remaining aircraft, noticeable only in the last period of the life of the CF-18 fleet.

- 41 -

CONCLUSIONS

82.   The CF-18 Structural Life Analysis Model was originally developed in 1989 and has been successfully used ever since as a support tool for various decisions regarding the structural life of the CF-18 fleet.   Since its initial development, the model underwent numerous modifications to keep it current with the changes occurring in the CF-18 composition and mode of operation.

83.   This report provides a comprehensive description of the latest version (March 2001) of the CF-18 SLAM. It contains a thorough description of the algorithms involved in the model, both from a conceptual perspective and from a programming perspective. Additionally, it contains a series of discussed examples of recent analyses that were performed using the model.   A crude analysis comparing CF-18 SLAM and its counterpart model developed by Bombardier is also included.

84.   It was found that, as a consequence of the present program of upkeep and modernization of 80 aircraft, a situation of concern will be encountered in the timeframe 2004 – 2010, when less than 70 CF-18 aircraft (on average) will be available for flying. The numbers can actually go as low as less than 50 aircraft available for flying in 2007, although the number of aircraft remaining in the fleet in the same year will be more than 80 units.

85.   Regarding the comparison between CF-18 SLAM and Bombardier's model, it has been concluded that, although there is a significant amount of overlap, the two models are really complementary (rather than similar). CF-18 SLAM was found to perform more like a forecasting tool, while its Bombardier counterpart was found to perform more like a scheduling tool. It is interesting to mention that one model's strengths represent the other model's weaknesses and the reverse, and therefore there is value in utilizing both models as tools to support the decision-making processes involved in the management of the CF-18 fleet.

- 42 -

LIST OF REFERENCES

1. Maj. D. Pelletier and P.E. Desmier, <u>Prediction of the CF-18 Fleet Size</u>, DAOR Staff Note 90/5, 1990

2. P.E. Desmier, <u>Sustaining the CF-18 Fleet into the 21st Century</u>, ORD Project Report 93/1, 1993

3. P.E. Desmier, <u>CF188 Structural Life Optimization</u>, ORD Project Report PR9704, 1997

4. D.W. Craig, <u>Peacetime Aircraft Attrition Process Modeling</u>, ORD Research Note 95/1, 1995

5. Palisade Corporation, <u>BestFit – User's Guide</u>, 1996

6. Algorithm AS III, <u>Applied Statistics</u> Vol.26, No.1, 1977

7. "Boeing to Upgrade CF-18s", <u>Wings</u> Issue 6, Vol. 41, pages 26-27, Dec/Jan/Feb 2000/2001

8. Fax from Capt. Jean Brosseau (DAEPM FT 2-3-2), 18 October 1999

ANNEX A
ORD PROJECT REPORT PR 2001/08
JUNE 2001

INPUT DATA FILES

TABLE A-1
FIRST INPUT DATA FILE: FLYING HOURS AND FLEI,
AS OF OCTOBER 31, 2000

Tail Number	Flying Hours	FLEI
701	1284	0 199
702	3175 3	0.554
703	3005 3	0 438
705	2563 6	0 364
706	3022 5	0 497
707	3405 5	0 511
708	3490 3	0 61
709	3334 9	0 527
710	3774.5	0 555
711	3953 7	0 51
712	3139 5	0 388
716	3658 2	0 523
718	3018 3	0 458
719	2146 3	0 315
720	3786.8	0.531
722	3625 2	0 556
723	3760 4	0 514
724	3563 3	0 436
725	3811 6	0.534
727	3420 9	0 621
728	3494 6	0 566
729	3333 2	0 599
730	3272 2	0 579
731	3318 8	0.383
732	3818 3	0 489
733	4491 2	0 483
734	3786 7	0 48
735	3544 9	0 529
736	3470 3	0 534
738	2577 2	0 4
739	3551 9	0 523
740	3225 4	0 543
741	3337 4	0 522

Tail Number	Flying Hours	FLEI
742	3485 5	0 531
743	3024 3	0 498
744	3717 9	0 503
745	3725 4	0 513
746	3708 7	0 522
747	3537 1	0 573
748	3829.2	0 52
749	3648 7	0 594
750	3393 5	0 621
751	3693	0 56
752	3904 6	0 549
753	3685 5	0 464
754	3891 4	0 528
756	3460 6	0 541
757	3833 4	0 52
758	3781 3	0 638
759	3857 2	0 542
760	3302 9	0 567
761	1814 9	0 224
762	2737 5	0.521
763	3653 5	0 559
764	2734	0 46
766	3272 2	0 564
767	3833 5	0 525
769	3760	0 542
770	3664 7	0 518
771	3551 6	0 471
774	3843 3	0 525
775	3307 8	0 546
776	3459 4	0 406
777	3597.7	0 437
778	3940.6	0.485
780	3843.4	0 473
781	4118 4	0 501
782	4037 4	0 524
783	3948 9	0.517
784	3208.2	0 486
785	3832 6	0 441
786	3570.2	0 529
787	4051 9	0 435
788	3864 4	0 505
789	3884 8	0 518
790	3979 1	0 461
791	3677 3	0 458
793	4346 1	0 508
794	3942 7	0 507

Tail Number	Flying Hours	FLEI
795	3578 6	0 513
796	3037 9	0 428
797	2262 6	0 345
798	3935 2	0 468
901	3392 6	0 51
902	3311 5	0 439
903	3488 3	0 417
904	2797 2	0 351
905	2976 7	0.383
906	3361 2	0 413
907	1875	0 149
908	3690 2	0 506
909	3511 1	0 501
910	3321 9	0 44
911	3363 9	0 492
912	4092 7	0.506
913	4074 4	0 539
914	4103.8	0 546
915	4148 9	0 56
916	4554 6	0 52
917	3372	0 373
918	3131 7	0 385
920	4188	0 564
921	4628 7	0 526
922	3915 1	0 524
923	4245 3	0 501
924	4034	0 499
925	4760 7	0 418
926	3893 2	0 465
927	3969 3	0 462
928	3750 7	0 446
929	3686 2	0 549
930	3522 6	0 466
931	3761 4	0 537
932	3965.2	0 494
933	3768 3	0 52
934	3764 6	0 529
935	3291 3	0 257
936	3890 1	0 518
937	3398 6	0 454
938	4085 3	0 425
939	3738 4	0 484
940	3725 8	0 419

TABLE A-2
SECOND INPUT DATA FILE: FLYING HOURS AT LAST PERIODIC
INSPECTION, AS OF 31 OCTOBER 2000

Tail Number	Flying Hours at Last Periodic Inspection
701	1047 0
702	2977 7
703	2858 5
705	2463 5
706	3008 3
707	3113 2
708	3269 3
709	3224 6
710	3569 3
711	3945.5
712	2722 9
716	3504 3
718	2599 6
719	1933 0
720	3717 3
722	3524 5
723	3439 4
724	3436 3
725	3423 3
727	3233 7
728	3410 6
729	3327 8
730	2965 5
731	3069 4
732	3394 6
733	4281 8
734	3352 7
735	3499 7
736	3350 9
738	2408 9
739	3472.7
740	2979.7
741	3192 9
742	3070 4
743	2716 8
744	3613 1
745	3451 8
746	3625 2
747	3423 4
748	3705 7
749	3621 7
750	3378 6
751	3335 4
752	3786 4

Tail Number	Flying Hours at Last Periodic Inspection
753	3487 6
754	3728 4
756	3457 7
757	3511 3
758	3767 5
759	3557 3
760	3198 0
761	1604 5
762	2732 7
763	3506 0
764	2413 6
766	2870 7
767	3685 7
769	3528 2
770	3230 0
771	3390 7
774	3646 4
775	2870 6
776	3419 2
777	3176 4
778	3908 1
780	3722 8
781	4055 2
782	3923 7
783	3862 4
784	2908 4
785	3813 8
786	3479.8
787	4021 6
788	3568 1
789	3841 6
790	3699 7
791	3292 3
793	3934 3
794	3751 0
795	3154 5
796	2894 2
797	1934 5
798	3758 3
901	3152 4
902	3147 1
903	3300 0
904	2644 8
905	2917 2
906	3133 0
907	1473 2
908	3648 0
909	3345 0
910	2979 3
911	3238 6

Tail Number	Flying Hours at Last Periodic Inspection
912	3843 8
913	4074 4
914	4026 1
915	3850 8
916	4172 9
917	3184 3
918	3036 5
920	3772 5
921	4208 3
922	3701 6
923	4157 6
924	3643 2
925	4616 8
926	3530 0
927	3554 0
928	3531 1
929	3683 8
930	3225 8
931	3599 1
932	3553 8
933	3690 2
934	3629 8
935	2904 2
936	3625 6
937	3398 6
938	3909 5
939	3712 8
940	3606 1

ANNEX B
ORD PROJECT REPORT PR 2001/08
JUNE 2001

MAINTAINING THE PROGRAM CURRENT

1.   The program requires a certain amount of periodic maintenance, to keep it current. As long as no major modifications in the composition or the *modus operandi* of the fleet occur, an adjustment of some of the model's parameters will suffice. The parameters that have to be periodically adjusted have already been mentioned in Part I of this report, along with the methods of adjustments. This Annex acts as a checklist, to help the analyst who has not already acquired a complete familiarity with the program, to implement scenarios and perform "what-if" analysis, as required.

2.   The following list contains the activities that must be performed by the analyst before implementing any new scenario:

a) Prepare (update) the file containing the latest FLEI and flying hours values for each aircraft of the fleet (see Annex A);

b) Prepare (update) the file containing the airframe hours at the last periodic inspection, for each aircraft of the fleet (see Annex A);

c) Make sure the two files above are correlated, in the sense that they both represent the same date. This date will constitute the "origin" on the time axis of the graphs, or, in other words, the present moment;

d) Update the function *"present_status_LEP"*, to incorporate in the model the latest data regarding the status of the implementation of the Life Extension Program;

e) Update the coefficients *a* and *b* that are involved in the attrition function *"crash_algorithm"*. A procedure to update these coefficients is given in Annex D;

f) Modify (if required) the fatigue rates to be used in the simulation (in the function *"allocate_fatigue"*), in accordance to the most recent values, such as indicated by the client; and

g) Modify (if required) the "base year" in the function "*statistic_results*", using the formula: *base year = present year + 1*;

3. The following list contains activities that must be performed by the analyst when a major event occurs, or periodically, at large intervals of time (e.g., one year):

a) In case a new crash occurs, the files containing the input data (i.e., FLEI and flying hours as well as flying hours at the last periodic inspection corresponding to each tail number) have to be modified accordingly;

b) In case a new crash occurs, particular care should be directed to the update of the coefficients *a* and *b* used in the attrition function, because it is in such situations that they suffer the most significant changes;

c) Yearly, the function "*third_line_maintenance*", implementing the Life Extension Program, or, more precisely, the three packages of structural modifications CP1, CP2 and CP3, should be updated. The way in which this function should be updated will be dictated by the reality of the time of update, concerning the LEP; and

d) Periodically, or if the policy of "flying the aircraft" changes in a significant way, the rules used for "control" in the function assigning flying hours to the aircraft should be reviewed and, eventually, updated to reflect the current situation.

4. Finally, in the event of major changes in the composition or concept of employment of the CF-18 aircraft, major modifications in the program are to be expected. The modular structure of the program and the liberal use of comments should make this updating process as easy and smooth as possible. However, if the changes to be implemented are beyond a certain level and the analyst is not yet fully comfortable with the program, developing a new implementation of the model should be envisaged.

ANNEX C
ORD PROJECT REPORT PR 2001/08
JUNE 2001

HANDLING VARIOUS PARAMETERS OF THE MODEL

Yearly Flying Rate

1.   The Yearly Flying Rate is one of the key factors involved in decisions regarding the management of the CF-18 fleet. Fortunately, it is also one of the parameters that is easiest to handle in the model. For convenience, the part of code concerning the YFR is reproduced below (Annex G contains the entire code listing):

```
void set_YFR ( )
{
      if (year = = 0)
      {
            yfr = YFR1;
      }
      else if ( (year > 0) && (year < 2) )
      {
            yfr = YFR2;
      }
      else
      {
            yfr = YFR3;
      }
}
```

2.   The code given above corresponds to a scenario in which the YFR varies during the years, taking one value (YFR1) during the first year, another value (YFR2) in the second year, and a third value (YFR3) from the third year after. If we would like a unique value for the YFR (say, YFR1) for the whole time horizon of the simulation, the function *set_YFR* would have to be modified in a very simple way. The new function would be:

```
void set_YFR ( )
{
      yfr = YFR1;
}
```

3.   In case one would need the second value (YFR2) to be kept over 4 years, beginning with the second year, one line of the function *set_YFR* of the first example would have to be changed. Only this line is reproduced hereafter:

```
...
else if ( (year > 0) && (year < 5) )
...
```

These examples should be sufficient to give the reader an idea of how YFR is treated in the model and that virtually any combination of YFR values can be modeled.

Tolerance

4.   This parameter describes the "precision" of the model, in relation to the YFR. A tolerance of three percent ensures that the YFR simulated by the model is within $\pm$ 3% of the desired YFR. Changing this parameter is straightforward. For example, if one would like to change the value of three percent by a value of five percent, the line

```
TOLERANCE 0.03
```

should be changed to the line

```
TOLERANCE 0.05
```

5.   The value of three percent has been found satisfactory. However, if there ever was a need to change it, one should be aware of the tradeoffs involved. By augmenting the value of this parameter, the precision of the model (with respect to the YFR) is decreased. By decreasing the value of the tolerance, the precision is increased, at the expense of a longer run time.

Simulation Horizon

6.   The time horizon for the simulation is given by the parameter MAX_YEARS, which can be changed as desired, in the same manner as exemplified for the TOLERANCE parameter.

Number of Iterations

7.   The number of iterations (NB_ITER) has to be big enough so that the results obtained from the model are statistically meaningful. A value of 1000 iterations has been found satisfactory. The user has the option to adjust it if required. However, a bigger value will increase the run time of the model.

Maximum Capacity at Bombardier

8.   Having limited resources, Bombardier has a maximum number of aircraft that it can work upon at the same time. Based on an induction plan provided by the plant for the period September 2000 – January 2003, the average number of aircraft in plant in any given month was found to be 12. However, since variations in the quantity of personnel assigned to the CF-18 structural modifications program are always possible, this parameter (CAP) should be updated regularly.   Also, by varying this parameter, one may determine, for instance, how the level of effort Bombardier is capable of providing impacts on the availability of aircraft for missions.

Duration of the Various Packages of Structural Modifications

9.   There are five parameters that define the length of the structural modifications packages.   They   are: duration_CP1,   duration_CP2,   duration_CP3, duration_CBR and duration_CBR_full.   The significance of the first three parameters have already been discussed in detail, the fourth parameter concerns solely the aircraft having tail number 759, while the fifth parameter refers to the three aircraft that are awaiting a centre barrel replacement (747, 749, 758). These parameters are more than likely to change as more information becomes available, after CP2 and CP3 are fully developed. Special care should be taken to update these parameters, since they have a major impact on the availability of aircraft.

ANNEX D
ORD PROJECT REPORT PR 2001/08
JUNE 2001

A PROCEDURE FOR UPDATING THE COEFFICIENTS "a" AND "b" INVOLVED IN THE ATTRITION FUNCTION "*crash_algorithm*"

1.   The coefficients a and b in the equation $L = ah^b$ (eqn. (1)) describing the attrition model, have to be updated each month, or each time a new crash occurs.

2.   To update these coefficients, one has to know the cumulative number of flying hours (for the fleet) corresponding to each of the past crashes, given in Table D-1, as well as the cumulative number of flying hours corresponding to the current month.

TABLE D-1
CF-18 CRASHES AND CUMULATIVE FLYING HOURS
UP TO 31 OCTOBER 2000

Crash Number	Cumulative Flying Hours
1	5973
2	17847
3	36766
4	63068
5	72521
6	89225
7	118000
8	158300
9	165369
10	167500
11	167500
12	172000
13	325291
14	327300
15	343040
15 (no new crashes)	434483 (as of 31 Oct 2000)

3.   The methodology to calculate the most up to date coefficients a and b has two phases:

    a) plot the cumulative number of crashes as a function of the cumulative number of flying hours; and

    b) using a curve fitting procedure, find the curve of the form $L = ah^b$ ("power" curve) that will best fit the graph plotted in phase (a).

The last point of the curve corresponds to the cumulative number of hours flown by the fleet to date (see Figure D1).

CF-18 Attrition
Data Accurate to 31 October 2000

f(x) = a * (x^6 581388E-1),
where a = exp(-5 763096E+0)
R^2 = 9 948870E-1

a = 0.00314137   b = 0.6581388

[THIS CURVE FIT IGNORES CRASHES 10-12]
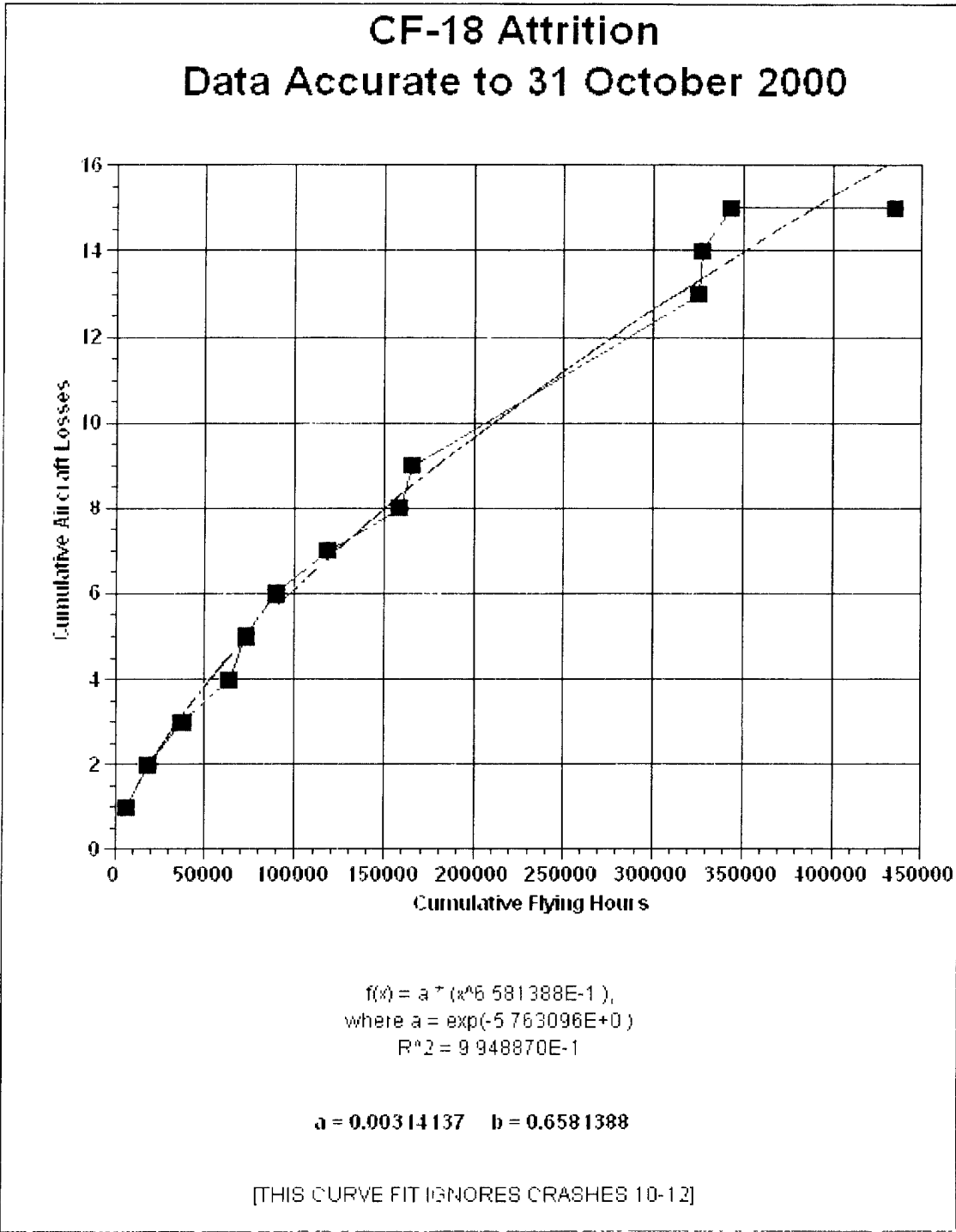
Figure D1. Computing coefficients *a* and *b*

ANNEX E
ORD PROJECT REPORT PR 2001/08
JUNE 2001

AN IMPLEMENTATION OF THE NORMAL DISTRIBUTION

1. Several times in the program there was a need to model variables as normally distributed. The function described here, called *"ppnd"* is devoted to this purpose. The function, inherited from the original version of the simulation (see Ref. 2 and 3), has not been changed in any way other than its translation from Fortran to C++.

2. For a better understanding on this algorithm, we recommend Ref. 6. The main idea behind this algorithm, as exposed in the cited reference, is that, although the integral defining the distribution function of the normal distribution cannot be evaluated by elementary methods, it can be represented in terms of the integral:

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{\frac{-u^2}{2}} \, du \qquad (E\text{-}1)$$

which is the distribution function of the normal distribution with mean 0 and variance 1, and which has been tabulated.

3. The algorithm implemented by the function *ppnd(p)* determines z, given the value of the integral $\Phi(z)$; i.e., it solves the equation (E-1) in the variable z. Then, using the substitution formula used to pass from F(x) to $\Phi(z)$, which is $z = (x-\mu)/\sigma$, the value of x can be obtained, or, in other words, one can actually assign a numerical value to a variable which was known to be normally distributed with a mean μ and a standard deviation σ.

ANNEX F
ORD PROJECT REPORT PR 2001/08
JUNE 2001

HOW TO INSTALL CF-18 SLAM

1.   The source code of CF-18 SLAM is provided on CD-ROM. The CD-ROM contains the whole set of input /output files used /produced by the model. as well. For installation, the use of compressed ("zip") files is recommended.

2.   To install CF-18 SLAM from the CD-ROM. the following steps should be followed:

   a) Ensure that the computer on which the installation is desired has WinZip and Microsoft Visual C++ version 5.0 or later, installed;

   b) Create a new folder that will contain CF-18 SLAM;

   c) Copy the folder *CF18forecasting.zip* to the new folder created;

   d) Rightclick on the copied file and choose the option "Extract to folder CF18forecasting";

   e) Start Visual C++;

   f) In the *File* menu, choose option "Open Workspace"; in the dialog box that appears, go to the location where the files were copied; select the file *CF18forecasting.dsw* and open it; this will bring up one of the files of CF-18 SLAM.

   g) In the *View* menu, choose "Workspace" and than doubleclick on *CF18forecastingDlg.cpp*; this file contains the source code for CF-18 SLAM (listed in Annex G);

   h) In the *Build* menu, choose "Rebuild All";

   i) After the rebuilding process is complete, choose "Execute CF18forecasting.exe" from the *Build* menu; this command initiates the execution of CF-18 SLAM; a dialog box appears, informing the user that the program may take a few minutes to execute; click on "Run" in this dialog box; during the execution of the program, this dialog box will remain on the screen; it disappears when the program finished execution;

F-2

j) The output files produced by the model can be found in the same folder that was created to host CF-18 SLAM.

3. Any modifications to the model should be effectuated in the file *CF18 forecastingDlg.cpp*, and should be followed by the usual procedure (compiling, rebuilding, debugging (eventually) and executing the program).

ANNEX G
ORD PROJECT REPORT PR 2001/08
JUNE 2001

THE VISUAL C++ CODE OF THE CF-18 SLAM

```
// CF18forecastingDlg.cpp : implementation file

#include "stdafx.h"
#include "CF18forecasting.h"
#include "CF18forecastingDlg.h"

#include "stdio.h"
#include "time.h"
#include "stdlib.h" //for rand() function
#include "math.h"

#define DIM_FLEET 122 //present dimension of the fleet//
#define TOLERANCE 0.03   //the YFR for the fleet must pe ensured within 3% tolerance
#define SIGMA 90 //flying hours for all aircraft are assumed to be normally
                 //distributed; SIGMA is the standard deviation associated to
                 //that distribution (derived from historical data)
#define MAX_YEARS 30 //the program calculates the parameters of interest (ex. remaining no. of aircraft each year), for the next 30 years
#define NB_ITER 1000 //number of iterations

#define YFR1 18300 // in case the YFR is not constant along the years. YFR1 will be the
#define YFR2 17800 // first one used (for the first years). YFR2 will be the one used for
#define YFR3 16000 // the subsequent years, etc. (if more than three YFR levels are needed, other variables may be introduced).

//the following 6 constants are all related to the 3rd line maintenance activities
#define CAP 12 //maximum capacity at Bombardier (maximum number of aircraft that can be worked upon in the same month)
#define duration_CP1 7 // duration of the first package of structural modifications, as estimated in Nov 2000
```

```c
#define duration_CP2 7 //duration of the second package of structural mods (in months)
#define duration_CP3 5 //duration CP3 (in months)
#define duration_CBR 12 //my guess for the duration of CBR plus selected mods for A/C 759
#define duration_CBR_full 19 //my guess for the duration of full CBR (one pass) plus selected mods, for A/C 747, 749, 758

#define DIM1 10
#define DIM2 30   // DIM1 and DIM2 are dimensions of some arrays

//Prototypes
void set_YFR();
void read_database();
void present_status_LEP(); //LEP = Life Extension Program (i.e. CP1 + CP2 + CP3)
int crash_algorithm();
void update_crash_data(int);
void second_line_maintenance();
void third_line_maintenance();
void third_line_maintenance_ver2(); //implements the same scenario as used by Bombardier
void hours_flown();
void allocate_fatigue();
void fatigued_out_algorithm();
void time_passes();
void update_FLEI ();
void statistic_results();
void ppnd();

enum yn {yes, no};
enum sd {single, dual};

//global variables

unsigned int limit1, limit2;
//limit1 = life limit for the early production A/C
//limit2 = life limit for the non early production A/C
int nb_it=0, year=0, month=0; //variables compteur pour le nombre d'iterations, les annees at les mois, respectivement
```

G-2

```
float yfr=YFR1;
int compteur=0;
int remaining_ac=0;
int i=0, j=0, k=0;
int ac_this_year=0; //counts the number of A/C in plant in the current year
float estimated_yfr=0;

FILE *output;

int crashed_ac[DIM1]; //store the tail numbers of A/C crashed during the current year
int replacement_ac[DIM1]; // store tail numbers of A/C replacing the ones crashed
int fatigued_ac[DIM2]; // store tail numbers of A/C fatigued out during the current year

float mean=0; // mean number of hours flown per year per aircraft (= yfr/nb.of aircraft)
int active_fleet=99; //dimension of the active fleet (may be 99, 87, or any integer between 80 and 87
int n_crashes=0;

float copy_begin[DIM_FLEET]; //working arrays
float copy_end[DIM_FLEET];
float copy_year_begin[DIM_FLEET];
float copy_year_end[DIM_FLEET];
float delta_flei[DIM_FLEET]; // amount of FLEI gained by an A/C in the current year
float correct_delta_flei[DIM_FLEET]; //correction of the above, in the case the A/C has fatigued out in the current year
float correct_hours[DIM_FLEET]; //correction of the number of allocated hours for the current year, in case the A/C has fatigued out (in the current year)

struct AIRCRAFT
{
    int tail_number;
    float hours_flown;
    float flei;
    float fatigue_rate;
    enum sd type; //type = single or dual//
    enum yn epa; // EPA = early production aircraft//
```

G-3

```
enum yn aete; // AETE = Aerospace Engineering Test Establishment//
enum yn crashed;
enum yn fatigued_out;
int second_line; //can be "0" or "1"; if "0", the aircraft is not in the second line maintenance;
                 //if "1", the aircraft is presently in the second line maintenance (for 1 month)
int third_line; //can be "0" or "1"; if "0", the aircraft is not in third line maintenance;
                //if "1", it is (in CP1 or CP2, whatever)
float last_inspection; //stores the number of hours flown by an aircraft at the moment it had undergone
                       //its last inspection in second line maintenance.
int waiting; //can be "0" or "1"; if "1", the aircraft is waiting to get into 3rd line maintenance
             //(i.e., it cannot fly anymore, but cannot go into the plant either because Bombardier is
             //at full capacity); if "0", it means that the A/C is not waiting It may fly or be in maintenance or whatever, but it is not waiting

//the next three fields have been introduced to implement the newest information (November 2000) regarding the
//way the structural modifications will be managed (i.e., 3 packages of mods to be done in no more than two passes through Bombardier
int program; //records the maintenance program that has been assigned to each aircraft, according to the following notation:
/*         = "1", if the aircraft will receive the first combination of mods (CP1 followed by CP2 and CP3 combined)
           = "2", if the aircraft will receive the second combination of mods (CP1 + CP2 followed by CP3)
           = "3" if the aircraft will receive the third combination (all mods in one pass)
           = "4", only for A/C 759 which has already received CP1 and is now waiting for CBR
           = "5", only for A/C 747, 749 and 758, who are waiting for CBR (one pass)
           = "0", if no combination has been assigned to the aircraft yet; */
int in_plant; //can be "0" or other integer number; if "0", the A/C is not currently in plant;
              //any other integer number (besides 1000, which is used for initialization) will represent the
              //number of months that the A/C still has to spent in third line maintenance (at Bombardier)
int first_phase; //it is related to the third line maintenance activities; it can be "0" if the first phase of any
                 //of the cobinations has not been begun; it is "1" if the first phase has begun;
                 //REM: For the combinations that have only one phase, this will be considered "phase 1"
int done; //can be "0" or "1"; if "1", the aircraft has received all mods that it was supposed to receive and therefore
          //doesn't have to go in plant for structural mods anymore in its remaining life; if "0", it still has to go in plant for mods.

};

AIRCRAFT fleet[DIM_FLEET].
```

G-4

```cpp
int results_nb_crashes[NB_ITER][MAX_YEARS]; // array used to keep the results regarding
                                            // the crashes

int results_nb_fatigued_out[NB_ITER][MAX_YEARS]; //results regarding the fatigued out A/C
int results_remaining_ac[NB_ITER][MAX_YEARS]; //results regarding the remaining A/C
int results_remaining_singles[NB_ITER][MAX_YEARS];
int results_remaining_duals[NB_ITER][MAX_YEARS];
int results_crashes[NB_ITER][MAX_YEARS];
float results_hours[NB_ITER][MAX_YEARS];
int results_nb_second_line[NB_ITER][12][MAX_YEARS]; //results regarding the number of A/C in second line maintenance
int results_nb_third_line[NB_ITER][12][MAX_YEARS]; //results regarding the number of A/C in third line maintenance
int results_nb_waiting[NB_ITER][12][MAX_YEARS]; //results regarding the number of A/C waiting to go in plant
                                                 //(don't fly anymore but cannot go in plant because of lack of capacity
                                                 // at Bombardier)
int results_nb_operational[NB_ITER][12][MAX_YEARS];//results regarding the number of A/C that are operational
                                                    //(A/C that are available for missions)
-------------------------------------------------------------------------
void CCF18forecastingDlg::OnRunButton()
{
        //MY CODE - START// (7 March 2000)

srand( (unsigned) time( NULL ) ); //seed the random number           //generator with the current time;


//main program//

for (nb_it=1; nb_it <= NB_ITER; nb_it++)
{
        read_database();
        present_status_LEP();
        for (year=0; year < MAX_YEARS; year++)
        {
                set_YFR();
                n_crashes = crash_algorithm();
                update_crash_data(n_crashes);
```

G-5

```c
    for (month=1; month <= 12; month++)
    {
        second_line_maintenance();
        third_line_maintenance();
        hours_flown();
        if (month == 1)
        {
            allocate_fatigue(); //a certain aircraft is allocated the same rate of fatigue for the whole year
        }
        update_FLEI ();
        time_passes();
    }
    fatigued_out_algorithm();
    }
    statistic_results();
    exit(0);
}

/***********************************************************
 *              void set_YFR ()
 *
 * This function sets the YFR;
 * In this case YFR varies in three levels (but it might be easily modified
 * to more or less levels, depending on the scenario to implement).
 ***********************************************************/

void set_YFR ()
{
    if (year == 0)
    {
        yfr = YFR1,
```

G-6

```
        }
        else if ( (year > 0) && (year < 2) )
        {
                yfr = YFR2;
        }
        else
        {
                yfr = YFR3;
        }
}

/*****************************************************
*
*       void read_database()
*
* This function reads two text files and initializes the array
* fleet[DIM_FLEET] with the read data.
* The first file contains information about the (last updated)
* values of the number of hours flown as well as the FLEI for
* each of the aircraft in the fleet.
* The second text file contains the number of hours flown on each
* airframe at the last periodic inspection conducted on that
* airframe.
*
* The program terminates if the database files cannot be open
* or if the number of data entries in these files does not match
* the number of aircraft in the fleet.
*****************************************************/

void read_database()
{
        FILE *f_data, *f_data2;
        char *name_file="oct_00.txt";  //latest FLEI values
```

G-7

```c
char *name_file2="last_insp_oct_00.txt"; //airframe hours when the last periodic inspection was done
unsigned int i;

f_data = fopen(name_file, "r");
if (f_data == NULL)
{
    printf("\n The indicated file cannot be open.");
    exit(1);
}

i = 0;
while (fscanf(f_data, "%d%f%f", &fleet[i].tail_number, &fleet[i].hours_flown,
    &fleet[i].flei) > 0)
{
    fleet[i].fatigue_rate = 0;

    //singles or duals//
    if ((fleet[i].tail_number > 700) && (fleet[i].tail_number < 800) )
    {
        fleet[i].type = single;
    }
    else if (fleet[i].tail_number >= 900)
    {
        fleet[i].type = dual;
    }

    //EPA//
    if (fleet[i].tail_number == 701 ||
        (fleet[i].tail_number >= 901 && fleet[i].tail_number <= 905))
    {
        fleet[i].epa = yes;
    }
```

G-8

```
        else
        {
                fleet[i].epa = no;
        }

        //AETE//
        if (fleet[i].tail_number == 701 || fleet[i].tail_number == 796
            || fleet[i].tail_number == 907 || fleet[i].tail_number == 917)
        {
                fleet[i].aete = yes;
        }
        else
        {
                fleet[i].aete = no;
        }

        fleet[i].crashed = no;
        fleet[i].fatigued_out = no;

        i = i+1;

    }
    fclose(f_data);

    //test to see if the number of data entries in the database file is matching
    //the number of aircraft in the fleet

    if ( i != DIM_FLEET)
    {
            printf("\n The file indicated  does not contain the"
                    " required number of data entries.");
            exit(1);

    }

    //open the file containing the nb. of hours on each airframe at the last inspection
```

G-9

```
f_data2 = fopen(name_file2, "r");
if (f_data2 == NULL)
{
        printf("\n The file <<last_inspection>> cannot be open.");
        exit(1);
}

i = 0;
while (fscanf(f_data2, "%d%f", &fleet[i].tail_number, &fleet[i].last_inspection) > 0)
{
        fleet[i].second_line = 0;
        fleet[i].third_line = 0;
        fleet[i].first_phase = 0;
        fleet[i].program = 0; //the A/C doesn't have a maintenance program (combinations 1 to 5) assigned to it yet;
        fleet[i].waiting = 0;
        fleet[i].in_plant = 0;
        fleet[i].done = 0;
        i = i+1;
}
fclose(f_data2);

if (i != DIM_FLEET)
{
        printf("\n The file <<last inspection >> does not contain"
                " the required number of entries.");

        exit(1);
}
```

```c
/*******************************************************************************
*
*       void present_status_LEP ()
*
* This function offers the possibility of feeding the model with the most up to
* date situation in what regards the status of the Life Extension Program (LEP).
* This function is closely associated with the third line maintenance activities.
*
* Eliminating this function from the program will have the effect of the model
* considering that no aircraft has received any package of structural modifications
* at time "zero" (i.e., the time when the model is executed)
*
* The function as it is here describes the situation as it is in January 2001.
* The forecast provided by the model is as accurate as possible at this time.
*
* However, if, at a later time (let's say, in one year from now), the function is
* not updated,  the model will still function but the forecast will not be the most
* accurate possible because it will not use all the information available. More
* precisely, it will use the information that was available in January 2001 and
* than it will use, for the year after, only ESTIMATED information (as predicted
* by th emodel itself), rather than the REAL data, that became available in the
* meantime.
*
* Therefore, updates of this function are recommended as soon as new information
* became available.
*
*******************************************************************************/

void present_status_LEP()
{
    unsigned int i=0;

    for (i=0; i<DIM_FLEET; i++)
    {
        //in January 2001, we know that 19 aircraft have already received CP1;
```

G-11

```
//the implications of this fact are the following: these aircraft can
//only be now "combination 1" aircraft; they can fly up to a FLEI of 0.645;
//CP1 being already finished, these aircraft are available for missions
//(i.e., we know that they are not in plant)
if ( (fleet[i].tail_number == 735) || (fleet[i].tail_number == 736)
     || (fleet[i].tail_number == 739) || (fleet[i].tail_number == 740)
     || (fleet[i].tail_number == 742) || (fleet[i].tail_number == 748)
     || (fleet[i].tail_number == 752) || (fleet[i].tail_number == 754)
     || (fleet[i].tail_number == 756) || (fleet[i].tail_number == 759)
     || (fleet[i].tail_number == 762) || (fleet[i].tail_number == 763)
     || (fleet[i].tail_number == 767) || (fleet[i].tail_number == 769)
     || (fleet[i].tail_number == 770) || (fleet[i].tail_number == 775)
     || (fleet[i].tail_number == 784) || (fleet[i].tail_number == 786)
     || (fleet[i].tail_number == 924) )
{

fleet[i].program = 1; //this aircraft can only be "combination 1"
fleet[i].first_phase = 1; //to indicate that the aircraft actually
                           //begun receiving some structural modification
fleet[i].third_line = 0; //to indicate that the aircraft is available to fly (not in plant)

}

}

}

/***************************************************************
*       CRASH ALGORITHM - modified from the original one (Dr. Paul Desmier's)
*
*       modification date: March 18, 1999
*
* return value:  nb_crashes = number of crashes in the current year
*
* The algorithm determines the number of crashes - Poisson Distribution
*
*       ========NEW FORMULAE========
***************************************************************/
```

```c
int crash_algorithm()
{
#define a 0.0031414
#define b 0.6581388   // (a, b = parameters of the "learning curve" -  31 Oct 2000)

double expected_losses_1=0, expected_losses_2=0, expected_losses=0;  // expected nb. of crashes up to the beginning of the current year,
                                                                     //up to the end of the current year and during the current year, respectively;

int nb_crashes=-1; // number of crashes
long int fact=1; //for the calculus of factorial(nb_crashes)
double partial_sum=1; // significant only for calculus purposes
double random=0; //for calculus purposes
double prob_nb_crashes=0; //probability of having a number of crashes occuring
                          //in the current year = "nb_crashes"
float fl_hours=0; // total number of flying hours for the fleet
int i=0; //variable compteur
int dim_fleet = 0; //fleet dimension


//calculate the dimension of the fleet
dim_fleet = 0;
for (i=0; i < DIM_FLEET; i++)
{
        if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) )
        {
                dim_fleet = dim_fleet + 1;
        }
}

if (dim_fleet <= 5) //when only a few aircraft are left, we allow no more crashes
{
        nb_crashes = 0;
}
```

G-13

```
else
{

//determine the total number of flying hours for the fleet (this number represents
//the number of flying hours accumulated at the beginning of the current year)

    fl_hours = 0;
    for(i=0; i<DIM_FLEET; i++)
    {
        fl_hours = fl_hours + fleet[i].hours_flown;
    }
    results_hours[nb_it-1][year] = fl_hours;

//determine the number of crashes - Poisson distribution

    expected_losses_1 = a * pow(fl_hours, b); //expected number of losses at the beginning of the currrent year

    expected_losses_2 = a * pow(fl_hours + yfr, b); //expected nb. of losses at the end of the current year

    expected_losses = expected_losses_2 - expected_losses_1;

Do_it_again:    random = rand() % 1000;
                random = random/1000; //"random" is a random number between 0 and 0.999

    nb_crashes = -1;
    partial_sum = 1;
    fact = 1;
    while (random < partial_sum)
    {
        nb_crashes = nb_crashes + 1;
        if (nb_crashes == 0)
        {
            fact = 1;
```

G-14

```
            }
        else
            {
            fact = fact * nb_crashes;
            }

        prob_nb_crashes = (pow(expected_losses,nb_crashes) * exp(-expected_losses))/fact;
        partial_sum = partial_sum - prob_nb_crashes;

    }
    if (nb_crashes > 5)
        {
        goto Do_it_again;
        }

    results_nb_crashes[nb_it-1][year] = nb_crashes;


    //determine the cumulative number of crashes up to the current year

    if (year == 0)
        {
        results_crashes[nb_it-1][year] = results_nb_crashes[nb_it-1][year];
        }
    else
        {
        results_crashes[nb_it-1][year] = results_crashes[nb_it-1][year-1] +
                                    results_nb_crashes[nb_it-1][year];
        }

    return(nb_crashes);

}
```

G-15

```
/********************************************************************
*     void update_crash_data( int nb_crashes)
*
* param: nb_crashes = nb. of crashes as determined by the function "crash_algorithm"
* If the number of crashes passed as a parameter to the function is > 0,
* the function will randomly choose the required number of A/C ( = nb_crashes)
* from the active A/C and will update the database (updates fleet.crashed[i]...)
*
* modification 18 nov. 1998: Single A/C crash much more often than dual A/C.
*                 (specifically, 90% of crashes are singles while only
*                  10% of crashes are duals)
*
********************************************************************/

void update_crash_data (int nb_crashes)
{
    int ran_nb=0;
    int s_crashed=0, d_crashed=0; //number of singles/duals that have already
                         //crashed
    int singles_left=0, duals_left=0; // number of singles /duals left in the fleet
    int next_crash=1; // = 1 implies that the next crash will be a single
                         // = 0 implies that the next crash will be a dual

    int i=0; //variable compteur


    //initialize arrays

    //update the database

    while (nb_crashes > 0)
    {
    //modification 18 nov. 98. 90% of crashes are singles and 10% are duals
```

G-16

```
//count number of singles and, respectively, duals that have already crashed
// and ALSO:
// verify how many single and, respectively, dual aircraft in the fleet
// are not crashed or fatigued out (this is necessary to avoid a situation in
// which the next crash would have to be a single but only dual aircraft
// are still left in the fleet, which will cause the program to go into
//an infinite loop

s_crashed = 0;
d_crashed = 0;
singles_left = 0;
duals_left = 0;

for(i=0; i<DIM_FLEET; i++)
{
    if (fleet[i].crashed == yes)
    {
        if (fleet[i].type == single)
        {
            s_crashed = s_crashed + 1;
        }
        else
        {
            d_crashed = d_crashed + 1;
        }
    }
    else  //count the nb. of singles/duals that could eventually crash (not fatigued out and not AETE)
    {
        if ( (fleet[i].aete == no) && (fleet[i].fatigued_out == no) )
        {
            if (fleet[i] type == single)
            {
                singles_left = singles_left + 1;
```

```
        }
        else
        {
            duals_left = duals_left + 1;
        }
    }

    //verify the proportion between singles and duals already crashed and
    //decide what (type) would be the current year crash(es)
    //if the proportion is more than 9 singles to 1 dual => the current
    //year crash will be a dual; if the proportion is less than 9 singles
    //to 1 dual, the current year crash will be a single;
    if (s_crashed >= 9 * d_crashed)
    {
        if (duals_left > 0)
        {
            next_crash = 0;  //next crash will be a dual
        }
        else
        {
            next_crash = 1; // if no more duals are left, next crash will have to be a single;
                            // we know that a single is still left in the fleet because we know
                            // that if the nb. of aircraft in the fleet is <= 5, nb_crashes
                            //cannot be different from 0
        }
    }
    else
    {
        if (singles_left > 0)
        {
            next_crash = 1;  //next crash will be a single
        }
```

G-18

```c
        else
        {
            next_crash = 0; // next crashe will have to be a dual, regardless of the proportion
                            // between singles and duals crashed, because no more singles are left
        }

        ~

ran_nb = rand() % DIM_FLEET;
if ( ( ( (next_crash == 1) && (fleet[ran_nb].type == single))
     || ((next_crash == 0) && (fleet[ran_nb].type == dual)) )
     && (fleet[ran_nb].crashed == no) &&          //if the A/C indicated by
     (fleet[ran_nb].fatigued_out == no) &&        //the random number was
                                                  //active, not already crashed
     (fleet[ran_nb].aete == no) )                 //or fatigued out
                                                  //and was not an AETE aircraft (which
                                                  //don't crash), then...
{
    nb_crashes = nb_crashes - 1;
    fleet[ran_nb].crashed = yes; //update the database
}

    ~

/**************************************************************************
*
*
*
* After every 400 flying hours, an aircraft should undergo an inspection and, eventually,
* some minor reparations that can be done on the base (2nd line maintenance). Usually, an
* aircraft stays in 2nd line maintenance for one month.
*
* This function is scanning the number of hours flown by each aircraft, and decides, each
* month, which A/C should be sent into 2nd line maintenace for the current month  The

void second_line_maintenance()
```

G-19

```
 * algorithm is simple: if the difference between the number of hours flown up to the current
 * month - and the number of hours flown when the A/C had undergone its last inspection
 * is >= 400, then the A/C is sent for inspection.
 *
 * NOTE 1: "400 hours" should be actually read "400 +/- 40" (there is a little bit of
 * flexibility) - numbers provided by Capt. Jean Brosseau
 *
 * NOTE 2: This (i.e., the decision to send an aircraft into 2nd line maintenance) should be
 * done BEFORE the process of assigning flying hours for the current month.
 * The idea is to see, what should I do with the aircraft in the current month - let it fly
 * (i.e., assign flying hours) or send it for the inspection (and therefore not assigning any
 * flying hours)?
 *
 * REM: Unlike the "third_line_maintenance" function, all aircraft are eligible to go into
 * second line maintenance, as long as they still fly.
 ******************************************************************************/

void second_line_maintenance()
{

    unsigned int i=0;
    unsigned int nb_second_line=0; //number of aircraft in second line maintenance in the current month

    nb_second_line = 0;
    for(i=0; i < DIM_FLEET; i++)
    {

        if ( (fleet[i]. crashed == no) && (fleet[i].fatigued_out == no)
            && (fleet[i].second_line == 0) && (fleet[i].third_line == 0)
            && (fleet[i].waiting == 0) )

        {

            if ( (fleet[i] hours_flown - fleet[i].last_inspection) >= 400 )
            {

                fleet[i].second_line = 1; //flag the aircraft as "in 2nd line maintenance"
                fleet[i].last_inspection = fleet[i].hours_flown; //update the "date"
```

```
            //(i.e., # of hours flown) of the last inspection
            nb_second_line = nb_second_line + 1;
        }

        }

    results_nb_second_line[nb_it-1][month-1][year] = nb_second_line;

    }

/****************************************************************************
 *
 *          (void) third_line_maintenance()
 *
 * This function implements the structural modifications program, as of November 10, 2000
 *
 * However, it should be noted that things didn't reach the "steady state" yet and therefore
 * further changes are more than likely. Anyway, for now, the situation is as follows:
 *
 * The structural modifications required to upgrade the A/C so that they will be able to fly until
 * they reach a FLEI of 1.0 have been grouped in three packages, called CP1, CP2 and CP3.
 * Aircraft cannot fly beyond the threshold of 0.52 unless they undergo these structural
 * modifications. Therefore, once they reach this threshold (or, actually, 0.515, in the
 * conservative approach that was taken) they will have to go to Bombardier. Because dismantling
 * and re-assembling the aircraft is a costly process, it has been decided that, although there are
 * three packages of modifications, an aircraft will go in plant at most twice. In this sense,
 * the following combinations have been retained (agreed upon by both DND and Bombardier):
 * (1) 1st pass = CP1; 2nd pass = CP2+CP3 (combined) (i.e., a total of 2 passes)
 * (2) 1st pass = CP1 + CP2; 2nd pass = CP3 (i.e., 2 passes)
 * (3) 1st pass = CP1 + CP2 + CP3 (i.e., only one pass)
 *
 * The first package (CP1) will be able to extend the fatigue life of the A/C up to 0.645.
 * CP2 will be able to able to extend the life up to 0.733 while CP3 will further extend the fatigue
 * life up to a FLEI of 1.0.
```

G-21

```
 *
 * The mods included in CP1 have already been completed from an engineering point of view.
 * The mods in CP2 are expected to be finished by 2002, while those in CP3 are supposed to be ready
 * by 2003.
 *
 * It has been considered that aircraft that are already passed 0.57, should not be elligible for
 * the first combination, but rather for combination (2). In the meantime (until CP2 is ready, these
 * aircraft are restricted from flying.
 *
 * Of course, Bombardier has a limited level of effort available. From the induction plan provided
 * by Bombardier, we decided that a good value to describe this limited work capacity would be a
 * maximum of 12 aircraft that can be worked upon simoultaneously (i.e., in any given month).
 *
 * The estimated durations of the three packages are as follows: CP1 = 7 months; CP2 = 7 months;
 * CP3 = 5 months. These are the best guesses avaialable right now (data provided by Capt. Brosseau).
 *
 * The assumptions above are valid for all aircraft, but four: A/C 759 has already received CP1 and
 * is now awaiting for a central barrel replacement plus selected CP2 and CP3 mods; No estimation
 * is available for the duration of this program; aircraft 747, 749 and 758 are also waiting for a
 * a central barrel replacement, which will be done simoultaneously with selected CP1, CP2 and CP3 mods.
 * Again, no estimation is available in what concerns the duration or the moment these programs will begin.
 * For our implementation purposes, it has been considered that the
 * durations are similar to the ones estimated for the rest of the aircraft.
 *
 * This function also keeps track of the number of aircraft "waiting" (i.e. aircraft that cannot be
 * accepted at Bombardier due either to the limited level of effort that Bombardier can provide, or
 * due to the fact that the structural modifications that they are waiting for are not ready yet.
 *
 ********************************************************************************/

void third_line_maintenance()
{

    unsigned int i=0;  //counting variable
```

```c
unsigned int nb_this_month=0; //counts the number of aircraft that are in plant in the current month
unsigned int nb_waiting=0;
unsigned int nb_operational=0;

//initialization at the beginning of the current month

nb_this_month = 0,

// count the number of aircraft already in plant for the current month
for (i=0; i < DIM_FLEET; i++)
{
    if (fleet[i].third_line == 1)
    {
        nb_this_month = nb_this_month + 1;
    };
};

for (i=0; i<DIM_FLEET; i++)
{
    //verify eligibility to go in plant //data valid November 2000
    if ( fleet[i].tail_number != 701 && fleet[i].tail_number != 702
    && fleet[i].tail_number != 703 && fleet[i].tail_number != 705
    && fleet[i].tail_number != 706 && fleet[i].tail_number != 707
    && fleet[i].tail_number != 708 && fleet[i].tail_number != 709
    && fleet[i].tail_number != 710 && fleet[i].tail_number != 711
    && fleet[i].tail_number != 712 && fleet[i].tail_number != 716
    && fleet[i].tail_number != 718 && fleet[i].tail_number != 719
    && fleet[i].tail_number != 720 && fleet[i].tail_number != 722
    && fleet[i].tail_number != 723 && fleet[i].tail_number != 724
    && fleet[i].tail_number != 725 && fleet[i].tail_number != 727
    && fleet[i].tail_number != 764 && fleet[i].tail_number != 901
    && fleet[i].tail_number != 902 && fleet[i].tail_number != 903
```

G-23

```
                && fleet[j].tail_number != 904 && fleet[j].tail_number != 905
                && fleet[j].tail_number != 906 && fleet[j].tail_number != 907
                && fleet[j].tail_number != 908 && fleet[j].tail_number != 909
                && fleet[j].tail_number != 910 && fleet[j].tail_number != 911
                && fleet[j].tail_number != 912 && fleet[j].tail_number != 913
                && fleet[j].tail_number != 914 && fleet[j].tail_number != 915
                && fleet[j].tail_number != 916 && fleet[j].tail_number != 920
                && fleet[j].tail_number != 921 && fleet[j].tail_number != 922
                && fleet[j].tail_number != 929 && fleet[j].tail_number != 931)
        }

        //give priority to aircraft already waiting to go in plant
        if (fleet[i].waiting == 1)
        {

                if (nb_this_month < CAP)
                {

                        fleet[i].waiting = 0;
                        nb_this_month = nb_this_month + 1;

                //decide which package of modifications the A/C is supposed to undergo
                if (fleet[i].program == 1)   //if the A/C is "combination 1"
                {

                        if (fleet[i].first_phase == 0)  //if the first phase didn't begin
                        {

                                fleet[i].in_plant = duration_CP1; //send A/C in plant for phase 1 (i.e., CP1)
                                fleet[i].third_line = 1; //mark the A/C as being in plant;
                                fleet[i].first_phase = 1; //mark the A/C as having begun the first phase;

                        }

                        else if (fleet[i].first_phase == 1)
                        {

                                fleet[i].in_plant = duration_CP2 + duration_CP3;
                                fleet[i].third_line = 1;
                                fleet[i].done = 1;

                        }

                }
```

G-24

```
if (fleet[i].program == 2)
{
    if (fleet[i].first_phase == 0) //if the first phase didn't begin
    {
        fleet[j].in_plant = duration_CP1 + duration_CP2; //send A/C in plant for CP1 and CP2

        fleet[j].third_line = 1;
        fleet[j].first_phase = 1;
    }
    else if (fleet[i].first_phase == 1)
    {
        fleet[j].in_plant = duration_CP3;
        fleet[j].third_line = 1;
        fleet[j].done = 1;
    }
}

if (fleet[i].program == 3)
{
    if (fleet[i].first_phase == 0)
    {
        fleet[j].in_plant = duration_CP1 + duration_CP2 + duration_CP3;
        fleet[j].third_line = 1;
        fleet[j].first_phase = 1;
        fleet[j].done = 1;
    }
}

if ( (fleet[i].program == 4) && (year >= 2) )//A/C 759 only; we assume CBR will be available in year = 2
{
    if (fleet[i].first_phase == 0)
    {
        fleet[j].in_plant = duration_CBR; //if this aircarft is waiting,
```

combined

```
                    fleet[i].third_line = 1; //it can only wait for CBR; CP1 already done
                    fleet[i].first_phase = 1;
                    fleet[i].done = 1;

            }

        }

        if ( (fleet[i].program == 5) && (year >= 2) ) //A/C 747, 749 and 758 only
        {
            if (fleet[i].first_phase == 0)
            {
                    fleet[i].in_plant = duration_CBR_full;
                    fleet[i].third_line = 1;
                    fleet[i].first_phase = 1;
                    fleet[i].done = 1;

            }

        }

    }

//scan the aircraft that have flown in the current month, assign a maintenance program to each of
//the aircraft that doesn't have any assigned yet, and find the candidates to go into third line maintenance, if any.

if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no)
    && (fleet[i].second_line == 0) && (fleet[i].third_line == 0)
    && (fleet[i].waiting == 0) && (fleet[i].done == 0) )
{

// separate the A/C 747, 749, 758 and 759, characterized by special mods (CBR) from the rest of the aircraft

if ( (fleet[i].tail_number != 747) && (fleet[i].tail_number != 749)
    && (fleet[i].tail_number != 758) && (fleet[i].tail_number != 759) )
{

//test conditions for aircraft "combination 1" (aircraft with a FLEI > 0.57 are not eligible
```

```
//for combination "1"

//identify the aircraft that will be "combination 1"

if ( (fleet[i].flei > 0.515) && (fleet[i].flei < 0.57) && (year == 0) )
{
    if (fleet[i].program == 0)
    {
        fleet[i].program = 1; //make this aircraft "combination 1"
    }
}

//among the aircraft "combination 1", identify those that have to go in plant for phase 1
if ( (fleet[i].program == 1) && (fleet[i].first_phase == 0) )
{
    //try to send this aircraft in plant
    if (nb_this_month < CAP)
    {
        fleet[i].in_plant = duration_CP1; //send in plant for the first package of modifications
        fleet[i].first_phase = 1; //mark that the aircraft has begun its first pass at Bombardier
        fleet[i].third_line = 1;
        nb_this_month = nb_this_month + 1;
    }
    else
    {
        fleet[i].waiting = 1; //mark the aircraft as "waiting"
    }
}

//among the aircraft "combination 1", identify those that have to go in plant for the second phase
if ((fleet[i].program == 1) && (fleet[i].flei >= 0.645) && (fleet[i].first_phase == 1) )
{
    if (nb_this_month < CAP)
    {
        fleet[i].in_plant = duration_CP2 + duration_CP3;
```

G-27

```
            fleet[i].third_line = 1;
            nb_this_month = nb_this_month + 1;
            fleet[i].done = 1; //mark the fact that this aircraft doesn't have to go in plant
                                //for structural mods never in its remaining life
        }
        else
        {
            fleet[i].waiting = 1;
        }
    }
}

//test conditions for aircraft "combination 2"

// identify the aircraft that will be "combination 2"
//("combination 2" will be all aircraft that have a FLEI > 0.57 in year = 0, plus
// the aircraft that, in year = 1, have a FLEI between 0.515 and 0.645 and had not yet
// been designated as "combination 1")

if ( (fleet[i].flei > 0.57) && (year == 0) && (fleet[i].first_phase == 0) )
{
    fleet[i].program = 2;
    fleet[i].waiting = 1; //mark the aircraft as waiting, because CP2 is not available until year = 1;
}

if ( (year >= 1) && (fleet[i].flei > 0.52) && (fleet[i].program == 0) )
{
    fleet[i].program = 2;
}

//among the aircraft "combination 2", identify those that have to go in plant for phase 1
if ( (year >= 1) && (fleet[i].program == 2) && (fleet[i].first_phase == 0) )
{
    if (nb_this_month < CAP)
    {
```

G-28

```
                    fleet[i].in_plant = duration_CP1 + duration_CP2;
                    fleet[i].third_line = 1;
                    fleet[i].first_phase = 1;
                    nb_this_month = nb_this_month + 1;
                }
            else
                {
                    fleet[i].waiting = 1;
                }
        }

//among the aircraft "combination 2", identify those that have to go in plant for phase 2
    if ( (year >= 1) && (fleet[i].flei >= 0.733) && (fleet[i].first_phase == 1) )
        {
            if (nb_this_month < CAP)
                {
                    fleet[i].in_plant = duration_CP3;
                    fleet[i].third_line = 1;
                    fleet[i].done = 1;
                    nb_this_month = nb_this_month + 1;
                }
            else
                {
                    fleet[i].waiting = 1;
                }
        }

//test conditions for aircraft "combination 3"
    //identify the aircraft that will be "combination 3"
        // (these aircraft are all aircraft that by the year CP3 is ready, didn't already begin another program)
        //(OBS: Even for the aircraft that have already been assigned a program but have not yet begin that program,
        // the initial program will be changed for "3", because this is cheaper)
    if ( (year == 2) && (fleet[i].first_phase == 0) )
        {
            fleet[i].program = 3;
```

```
// among aircraft "combination 3", identify those that are due to be sent in plant
    if ( (year >= 2) && (fleet[i].flei > 0.52) && (fleet[i].program == 3) )
    {
        if (nb_this_month < CAP)
        {
            fleet[i].in_plant = duration_CP1 + duration_CP2 + duration_CP3;
            fleet[i].third_line = 1;
            fleet[i].first_phase = 1;
            fleet[i].done = 1;
            nb_this_month = nb_this_month + 1;
        }
        else
        {
            fleet[i].waiting = 1;
        }
    }

else    //the case of A/C 747, 749, 758 and 759
{
    if (fleet[i].tail_number == 759)
    {
        fleet[i].program = 4;
        if ( fleet[i].flei >= 0.645 )
        {
            if ( (nb_this_month < CAP) && (year >=2 ) )  //we asume CBR not available until year = 2
            {
                fleet[i].in_plant = duration_CBR;
                fleet[i].third_line = 1;
                fleet[i].first_phase = 1;
                fleet[i].done = 1;
                nb_this_month = nb_this_month + 1;
            }
        }
    }
```

G-30

```
                }
                else
                {
                    fleet[j].waiting = 1;
                }
            }

            if ( ( (fleet[i].tail_number == 747) || (fleet[i].tail_number == 749)
                || (fleet[i].tail_number == 758) )
            {
                fleet[j].program = 5;
                if (fleet[j].flei >= 0.515)
                {

                    if ( (nb_this_month < CAP) && (year >= 2) ) //same assumption as above
                    {

                        fleet[j].in_plant = duration_CBR_full;
                        fleet[j].third_line = 1;
                        fleet[j].first_phase = 1;
                        fleet[j].done = 1;
                        nb_this_month = nb_this_month + 1;

                    }
                    else
                    {

                        fleet[j].waiting = 1;

                    }
                }
            }
```

//count the number of aircraft that are in "waiting" status in the current month:

```
nb_waiting = 0;
for (i=0; i < DIM_FLEET; i++)
```

G-31

```
        if (fleet[i].waiting == 1)
        {
            nb_waiting = nb_waiting + 1;
        }
    }

    results_nb_third_line[nb_it-1][month-1][year] = nb_this_month;
    results_nb_waiting[nb_it-1][month-1][year] = nb_waiting;

    //count no. of operational aircraft in the current month ("operational aircraft" are the aircraft
    //that are available for missions
    nb_operational = 0;
    for (i=0; i <= DIM_FLEET; i++)
    {
        if (fleet[i].tail_number != 764)    //764 is B Cat. Damage
        {
            if ((fleet[i].crashed == no) && (fleet[i].fatigued_out == no)
            && (fleet[i].waiting == 0) && (fleet[i].second_line == 0) && (fleet[i].third_line == 0))
            {
                nb_operational = nb_operational + 1;
            }
        }
    }

    results_nb_operational[nb_it-1][month-1][year] = nb_operational;
}
```

G-32

```
/*******************************************************************
 *
 *        void third_line_maintenance_ver2 ()
 *
 * This version of the function "third_line_maintenance" has been created with the purpose
 * of being able to implement a certain scenario that has ben used by Bombardier in their
 * analyses.
 * In this scenario, all aircraft in the fleet receive the structural modifications in
 * two passes through Bombardier. First, they receive CP1, which permits them to fly until
 * a FLEI threshold of 0.645 is reached. Then, they receive "CP1 complement" (which is
 * equivalent to CP2 + CP3 in the first version of this function).
 *
 * Note that "CP1 complement" will be available in Jan 2004.
 *
 * All other assumptions remain the same.
 *
 *******************************************************************/

void third_line_maintenance_ver2()
{

    unsigned int i=0;  //counting variable
    unsigned int nb_this_month=0; //counts the number of aircraft that are in plant in the current month
    unsigned int nb_waiting=0;
    unsigned int nb_operational=0;


    //initialization at the beginning of the current month

    nb_this_month = 0;
```

G-33

```
// count the number of aircraft already in plant for the current month
for (i=0; i < DIM_FLEET; i++)
{
        if (fleet[i].third_line == 1)
        {
                nb_this_month = nb_this_month + 1;
        };
};

for (i=0; i<DIM_FLEET; i++)
{
        //verify eligibility to go in plant //data valid November 2000
        if ( fleet[i].tail_number != 701 && fleet[i].tail_number != 702
        && fleet[i].tail_number != 703 && fleet[i].tail_number != 705
        && fleet[i].tail_number != 706 && fleet[i].tail_number != 707
        && fleet[i].tail_number != 708 && fleet[i].tail_number != 709
        && fleet[i].tail_number != 710 && fleet[i].tail_number != 711
        && fleet[i].tail_number != 712 && fleet[i].tail_number != 716
        && fleet[i].tail_number != 718 && fleet[i].tail_number != 719
        && fleet[i].tail_number != 720 && fleet[i].tail_number != 722
        && fleet[i].tail_number != 723 && fleet[i].tail_number != 724
        && fleet[i].tail_number != 725 && fleet[i].tail_number != 727
        && fleet[i].tail_number != 764 && fleet[i].tail_number != 901
        && fleet[i].tail_number != 902 && fleet[i].tail_number != 903
        && fleet[i].tail_number != 904 && fleet[i].tail_number != 905
        && fleet[i].tail_number != 906 && fleet[i].tail_number != 907
        && fleet[i].tail_number != 908 && fleet[i].tail_number != 909
        && fleet[i].tail_number != 910 && fleet[i].tail_number != 911
        && fleet[i].tail_number != 912 && fleet[i].tail_number != 913
        && fleet[i].tail_number != 914 && fleet[i].tail_number != 915
        && fleet[i].tail_number != 916 && fleet[i].tail_number != 920
        && fleet[i].tail_number != 921 && fleet[i].tail_number != 922
        && fleet[i].tail_number != 929 && fleet[i].tail_number != 931)
```

G-34

```
//give priority to aircraft already waiting to go in plant
if (fleet[j].waiting == 1)
{

    if (nb_this_month < CAP)
    {

        fleet[j].waiting = 0;
        nb_this_month = nb_this_month + 1;

        //decide if the A/C is supposed to undergo CP1 or "CP1 complement"
        if (fleet[j].program == 1)
        {

            if (fleet[j].first_phase == 0)  //if the first phase didn't begin
            {

                fleet[j].in_plant = duration_CP1; //send A/C in plant for phase 1 (i.e., CP1)
                fleet[j].third_line = 1; //mark the A/C as being in plant;
                fleet[j].first_phase = 1; //mark the A/C as having begun the first phase;

            }
            else if (fleet[j].first_phase == 1)
            {

                fleet[j].in_plant = duration_CP2 + duration_CP3;  //"CP1 complement"
                fleet[j].third_line = 1;
                fleet[j].done = 1;

            }

        }

        if ( (fleet[j].program == 4) && (year >= 2) )//A/C 759 only: we assume CBR will be available in year = 2
        {

            if (fleet[j].first_phase == 0)
            {

                fleet[j].in_plant = duration_CBR; //if this aircarft is waiting,
                fleet[j].third_line = 1; //it can only wait for CBR; CP1 already done
                fleet[j].first_phase = 1;
```

G-35

```
                    fleet[i].done = 1;
                }
            }
        }

        if ( (fleet[i].program == 5) && (year >= 2) ) //A/C 747, 749 and 758 only
        {
            if (fleet[i].first_phase == 0)
            {
                fleet[i].in_plant = duration_CBR_full;
                fleet[i].third_line = 1;
                fleet[i].first_phase = 1;
                fleet[i].done = 1;
            }
        }
    }
}

//scan the aircraft that have flown in the current month, assign a maintenance program to each of
//the aircraft that don't have any assigned yet, and find the candidates to go into third line maintenance, if any.

if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no)
    && (fleet[i].second_line == 0) && (fleet[i].third_line == 0)
    && (fleet[i].waiting == 0) && (fleet[i].done == 0) )
{

    // separate the A/C 747, 749, 758 and 759, characterized by special mods (CBR) from the rest of the aircraft

    if ( (fleet[i] tail_number != 747) && (fleet[i].tail_number != 749)
        && (fleet[i].tail_number != 758) && (fleet[i] tail_number != 759) )
    {

        if ( (fleet[i].flei > 0.515) )
        {
```

```
if (fleet[i].program == 0)
{
    fleet[i].program = 1; //make this aircraft "combination 1"
}

//among the aircraft "combination 1", identify those that have to go in plant for phase 1
if ( (fleet[i].program == 1) && (fleet[i].first_phase == 0) )
{
    //try to send this aircraft in plant
    if (nb_this_month < CAP)
    {
        fleet[i].in_plant = duration_CP1; //send in plant for the first package of modifications
        fleet[i].first_phase = 1; //mark that the aircraft has begun its first pass at Bombardier
        fleet[i].third_line = 1;
        nb_this_month = nb_this_month + 1;
    }
    else
    {
        fleet[i].waiting = 1; //mark the aircraft as "waiting"
    }
}

//among the aircraft "combination 1", identify those that have to go in plant for the second phase
// "CP1 complement" (the second phase) is available beginning Jan 2004
if ((fleet[i].program == 1) && (fleet[i].flei >= 0.645) && (fleet[i].first_phase == 1) && (year >= 3))
{
    if(nb_this_month < CAP)
    {
        fleet[i].in_plant = duration_CP2 + duration_CP3;
        fleet[i].third_line = 1;
        nb_this_month = nb_this_month + 1;
        fleet[i].done = 1; //mark the fact that this aircraft doesn't have to go in plant
                           //for structural mods never in its remaining life
    }
}
```

G-37

```
                else
                {
                        fleet[i].waiting = 1;
                }
        }

else    //the case of A/C 747, 749, 758 and 759
{
        if (fleet[i].tail_number == 759)
        {
                fleet[i].program = 4;
                if ( fleet[i].flei >= 0.645 )
                {
                        if ( (nb_this_month < CAP) && (year >=2 ) )  //we asume CBR not available until year = 2
                        {
                                fleet[i].in_plant = duration_CBR;
                                fleet[i].third_line = 1;
                                fleet[i].first_phase = 1;
                                fleet[i].done = 1;
                                nb_this_month = nb_this_month + 1;
                        }
                        else
                        {
                                fleet[i].waiting = 1;
                        }
                }
        }
        if ( (fleet[i].tail_number == 747) || (fleet[i].tail_number == 749)
                || (fleet[i].tail_number == 758) )
        {
                fleet[i].program = 5;
                if (fleet[i] flei >= 0.515)
                {
```

G-38

```
if ( (nb_this_month < CAP) && (year >= 2) ) //same assumption as above
{
    fleet[i].in_plant = duration_CBR_full;
    fleet[i].third_line = 1;
    fleet[i].first_phase = 1;
    fleet[i].done = 1;
    nb_this_month = nb_this_month + 1;
}
else
{
    fleet[i].waiting = 1;
}

//count the number of aircraft that are in "waiting" status in the current month;

nb_waiting = 0;
for (i=0; i < DIM_FLEET; i++)
{
    if (fleet[i].waiting == 1)
    {
        nb_waiting = nb_waiting + 1;
    }
}

results_nb_third_line[nb_it-1][month-1][year] = nb_this_month;
results_nb_waiting[nb_it-1][month-1][year] = nb_waiting;
```

G-39

```
//count no. of operational aircraft in the current month ("operational aircraft" are the aircraft
//that are available for missions
nb_operational = 0;
for (i=0; i <= DIM_FLEET; i++)
{
    if (fleet[i].tail_number != 764)  //764 is B Cat. Damage
    {
        if ((fleet[i].crashed == no) && (fleet[i].fatigued_out == no)
        && (fleet[i] waiting == 0) && (fleet[i].second_line == 0) && (fleet[i].third_line == 0))
        {
            nb_operational = nb_operational + 1;
        }
    }
}

results_nb_operational[nb_it-1][month-1][year] = nb_operational;

}

/************************************************************************************
*                                                                                  *
*                       void hours_flown()                                         *
*                                                                                  *
* Purpose: to assign flying hours for the current month, to each of the aircraft in the fleet.
*          Also, it must ensure that the monthly flying rate (MFR) is reached within
*          a given tolerance (3%)
*                                                                                  *
*************************************************************************************/

void hours_flown(void)
{

int i=0; //counting variables
float z=0;
int rdm=0;
```

G-40

```
int twice_the_mean=0;
float MFR_sim=0, //MFR obtained from the simulation in the current month;
int flying_ac_this_month=0; //number of aircraft that are actually flying in the current month

// keep a copy of the number of flying hours for each aircraft, at the beginning of the
// current month (i.e., at the beginning of the process of allocating hours)

for (i=0; i< DIM_FLEET; i++)
{

        copy_begin[i] = fleet[i].hours_flown;

}

for (i=0; i < DIM_FLEET; i++)
{

        if (month == 1)
        {
                copy_year_begin[i] = fleet[i].hours_flown;
        }
        if (month == 12)
        {
                copy_year_end[i] = fleet[i].hours_flown;
        }
}

// count the number of aircraft that are flying in the current month

flying_ac_this_month = 0;
for (i=0; i < DIM_FLEET; i++)
{

        if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) && (fleet[i].second_line == 0)
                && (fleet[i].third_line == 0) && (fleet[i].waiting == 0) )
        {

                flying_ac_this_month = flying_ac_this_month + 1;
```

```
        }

// determine the mean of hours flown per aircraft in the current month

if (flying_ac_this_month == 0)
    {
        goto alfa;
    }
else
    {
        mean = yfr/12/flying_ac_this_month;
    }
```

// The following part of the algorithm is based on the following considerations:
//
// Data analyzed = year 1999 (i.e., the number of hours flown monthly by each of the aircraft in the fleet
// = 122 A/C * 12 months = 1464 entries)
//
// The "BestFit" software decided that no distribution was a good enough fit for our data (all distributions
// have been rejected). Therefore, we couldn't make the assumption that the monthly flying hours/aircraft follow
// a certain distribution => we had to look for other ways to implement the randomness of the
// number of flying hours/aircraft/month.
//
// Solution: a somewhat "controlled" randomness. The rules for "control" reflect the reality of the year 1999
// (YFR = 21394 hrs), asssumed to be a "typical year" for the fleet. "Typical" not in the sense of the value
// of YFR, but more in the sense of the way the YFR is distributed among the aircraft of the fleet.
// If the operations would dramatically change in the future, other control rules should be implemented.
// A periodical revision of these rules (once a year ?) is recommended.
//
// 1999 data showed that 93.5% of the aircraft that have flown in 1999 had a number of hours flown by month between
// 1 and 43 hours, while the mean of the monthly number of hours flown was 21.74 for 1999. The rest of 6.5% of
// the aircraft have flown a number of hours between 44 and 120.
// Following these observations, it has been chosen to assign to ALL aircraft in the fleet

G-42

```
// a number of flying hours randomly selected in the interval [1, twice the mean]. I.e., we use for 100% of
// the aircraft the "model" of the 93.5%. The fact that we never assign (in any month) to an aircraft a "big" number
// of flying hours (such as for the 6.5% in our sample data) does not represent a limitation of the simulation,
// given the facts that the horizon simulation is 30 years and 500 iterations are performed.
// Also, we observe that no aircraft will have "big" numbers of flying hours assigned but ocasionnally.
//
// To generate the random numbers, the C++ "rand()" function is used.

// Of course, the total number of hours flown assigned in the current year must "equal" the YFR
// (with a certain tolerance = 3%)


//allocate hours for the current month

Try_again: MFR_sim = 0;   //initialize the monthly flying rate (MFR) for the current month;
        for (i=0; i < DIM_FLEET; i++)
        {

            if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) &&
                    (fleet[i].second_line == 0) && (fleet[i].third_line == 0) && (fleet[i].waiting == 0) )
            {

                //if the aircraft is flying in the current month, some flying hours
                //are tentatively assigned; "tentatively" because in case the MFR is not reached with
                //the desired tolerance, a new iteration is done and a different number of flying hours may be assigned

                twice_the_mean = (int) floor(2*mean);
                rdm = rand() % twice_the_mean;
                fleet[i].hours_flown = copy_begin[i] + rdm; //assign hours

                MFR_sim = MFR_sim + rdm;

            }

        }

//check if the monthly flying rate has been reached within the desired tolerance
//if not, a new iteration is performed and a new set of flying hours will be assigned
```

G-43

```
                if ( (MFR_sim < (yfr/12 - TOLERANCE*(yfr/12)) ) || (MFR_sim > (yfr/12 + TOLERANCE*(yfr/12)) ) )
                {
                        goto Try_again;
                }
        }
        else
        {
alfa:   for(i=0; i<DIM_FLEET; i++)
                {
                        copy_end[i] = fleet[i].hours_flown;
                }
        }
}


/******************************************************************************************
*
*               void time_passes (void)
*
* This function is updating the data regarding how much time the aircraft are in plant.
* More precisely, it is decreasing, each month, the number of munths that aircraft still have
* to spend in plant by one.
*******************************************************************************************/

void time_passes ()
{
int i=0; //counting variable

        for (i=0; i < DIM_FLEET; i++)
        {
                if (fleet[i].second_line == 1 )
```

G-44

```
            }

            fleet[i].second_line = 0; //if the aircraft was in second line maintenance, at the end of the current month
                                      //it will be operational again (since 2nd line maint duration = 1 month)
        }

        if (fleet[i].third_line == 1)
        {
            fleet[i].in_plant = fleet[i].in_plant - 1;
            if (fleet[i].in_plant == 0)
            {
                fleet[i].third_line = 0; //mark aircraft as operational again (i.e., not in third line maintenance
            }
        }
    }

}


/*****************************************************************************
*
*       double ppnd(p)
*
*
*       Ref:  Algorithm AS III Applied Statistics (1977) Vol.26, No.1   (in Fortran)
*
* Some theory:
* The integral defining the distribution function of the normal distribution
* cannot be evaluated by elementary methods, but it can be represented in terms
* of the integral:
*
* which is the distribution function of the normal distribution with mean 0 and
* variance 1 (and which has also been tabulated).
*
* The algorithm implemented by the function ppnd (p) determines  z, given the
* value of the integral. (It solves the equation (*) in the variable z).
```

```
 * Then, using the formula : z = (x-mean)/st_dev (which is the substitution used
 * to pass from F(x) to F(z) ), we can obtain the value of x - i.e., we can
 * actually assign a numerical value for a variable we knew it was normally
 * distributed with a certain mean (mean) and standard deviation (st_dev).
 *
 * param: p ( value of the integral)
 * return value: z
 *
 * Real version for eps = pow(2, -31)
 * The hash sums are the sums of the moduli of the coefficients. They have no
 * inherent meanings but are included for use in checking transcriptions.
 ***************************************************************************/

double ppnd(double p)
{
    double zero=0, half=0.5, one=1, split=0.42;
    double q, r, z;
    int ifault=0; //fault indicator; if it is "1" something had gone wrong
    long double a0=2.50662823884;
    long double a1=-18.61500062529;
    long double a2=41.39119773534;
    long double a3=-25.44106049637;
    long double b1=-8.47351093090;
    long double b2=23.08336743743;
    long double b3=-21.06224101826;
    long double b4=3.13082909833;

    // hash sum ab 143.70383558076

    long double c0=-2.78718931138;
    long double c1=-2.29796479134;
    long double c2=4.85014127135;
    long double c3=2.32121276858;
    long double d1=3.54388924762;
```

G-46

```
long double d2=1.6370678l897;

//hash sum cd 17.43746520924

ifault = 0;
q = p - half;
if ( abs(q) <= split )
{
    r = q*q;
    z = q*(((a3*r + a2)*r +a1) *r +a0) / ((((b4*r + b3)*r +b2)*r +b1) *r + one);
}
else
{
    r = p;
    if (q > 0)
    {
        r = one - p;
    }
    if (r > 0)
    {
        r = sqrt(exp(1/r));
        z = (((c3*r + c2)*r +c1)*r +c0) / ((d2*r +d1) *r + one);
        if (q < 0)
            z = -z;      //sign of z is equal to the sign of q
    }
    else
    {
        ifault = 1;
        z = zero;
    }
}
return (z);
```

G-47

```
/*******************************************************************************
 *
 *              void  allocate_fatigue(void)
 *
 * Determines the fatigue rate and the FLE for each aircraft.
 * Assumes fatigue rates are normally distributed.
 * This function is called once each year of the simulation; i.e., the implicit
 * assumption is that each aircraft will have assigned the same fatigue rate for
 * the whole current year (the fatigue rates are not assigned monthly!!!)
 * Assumes fatigue rates are the same regardless of the type of aircraft
 * (i.e., single, dual, AETE, EPA).
 *
 *******************************************************************************/

void allocate_fatigue(void)
{

        double rd=0, zzz=0;
        int i=0;

        //initialization at the beginning of the year
        for (i=0; i<DIM_FLEET; i++)
        {
                fleet[i].fatigue_rate = 0;
        }

        //allocate fatigue

        for (i=0; i<DIM_FLEET; i++)   //for each A/C not crashed and not fatigued out
        {
                if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) )
                {
```

G-48

```c
        while ( (fleet[i].fatigue_rate < 0.008)
                || (fleet[i].fatigue_rate > 0.32) )
            //place bounds on the fatigue rates, based on the 31 Dec 97 data
        {
            rd = (float) (rand() % 1000);
            rd = rd/1000;
            zzz = ppnd(rd);

            fleet[i].fatigue_rate = (float) (0.135 + zzz*0.06);
        }
    }
}

/***********************************************************************
 *
 *
 * This function updates the FLEI of each aircraft, according to the number of
 * hours flown by each aircraft in the current month (and, of course, to the
 * fatigue rate assined for each aircraft for the current year).
 *
 ***********************************************************************/

void update_FLEI ()
{
    //calculate FLE
    for (i=0; i<DIM_FLEET; i++)  //for each A/C not crashed and not fatigued out
    {
        if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) )
        {
            delta_flei[i] = (fleet[i].fatigue_rate * (fleet[i].hours_flown - copy_begin[i]))/1000;
```

G-49

```
            // FLE gained in the current month by the aircraft i
            fleet[i].flei = fleet[i].flei + delta_flei[i]; //update database

        }

    }

}

/***************************************************************************
 *
 *          void fatigued_out_algorithm()
 *
 * Identifies (and counts) the A/C that have fatigued out during the year.
 *
 * A "fatigued out" aircraft can be defined in two ways:
 *              1. in terms of "equivalent flying hours";
 * or
 *              2. in terms of the FLE index;
 * The equivalence between (1) and (2) (both of them measures of the fatigue
 * accumulated by the aircraft) is straightforward: the design value of
 * 6000 EFH (for the CF-18) is equivalent to a FLE index of 1.
 * The equation correlating the two is:  EFH(j) = FLE(j) * 6000,
 *
 * In this algorithm, the measure used to determine if an aircraft is fatigued out
 * or not is the FLE index.
 ***************************************************************************/

void fatigued_out_algorithm ()
{
        unsigned int nb_fatigued_out=0;
        unsigned int i=0, j=0, k=0;
        int remaining_ac=0, remaining_singles=0, remaining_duals=0;
```

```
/******************* MODIFICATION *******************************************
Some aircraft are eliminated from the fleet when they need important
structural modifications.
They are eliminated (flagged as "fatigued_out") when they reach a level of
fatigue exceeding a given threshold (0.56 in this case, for most of the aircraft)
***************************************************************************/

/* START MODIFICATION */

for(i=0; i<DIM_FLEET; i++)
{
//version: fleet reduced gradually up to 80 A/C - data valid November 2000 (Capt. Jean Brosseau)
if ( (year == 0) &&
        (fleet[i].tail_number == 722 || fleet[i].tail_number == 764) )
{
    fleet[i].fatigued_out = yes; //these aircraft are already retired
}

if (( fleet[i].tail_number == 701 || fleet[i].tail_number == 702
    || fleet[i].tail_number == 703 || fleet[i].tail_number == 705
    || fleet[i].tail_number == 706 || fleet[i].tail_number == 707
    || fleet[i].tail_number == 710 || fleet[i].tail_number == 711
    || fleet[i].tail_number == 712 || fleet[i].tail_number == 718
    || fleet[i].tail_number == 719 || fleet[i].tail_number == 723
    || fleet[i].tail_number == 724
    || fleet[i].tail_number == 906 || fleet[i].tail_number == 907
    || fleet[i].tail_number == 908 || fleet[i].tail_number == 909
    || fleet[i].tail_number == 910 || fleet[i].tail_number == 911
    || fleet[i].tail_number == 912 || fleet[i].tail_number == 913
    || fleet[i].tail_number == 914 || fleet[i].tail_number == 915
    || fleet[i].tail_number == 916 || fleet[i].tail_number == 920
    || fleet[i].tail_number == 921 || fleet[i].tail_number == 922
    || fleet[i].tail_number == 929 || fleet[i].tail_number == 931)
```

```
                        && (fleet[i].flei >= 0.56) )

                    fleet[i].fatigued_out = yes;

            }
        }
    else if ( (fleet[i].tail_number == 901 || fleet[i].tail_number == 902
        || fleet[i].tail_number == 903 || fleet[i].tail_number == 904
        || fleet[i].tail_number == 905)
        && (fleet[i].flei >= 0.52) )

            fleet[i].fatigued_out = yes;

        }
    }
    else if ( (fleet[i].tail_number == 709 || fleet[i].tail_number == 716
        || fleet[i].tail_number == 720 || fleet[i].tail_number == 725)
        && (fleet[i].flei >= 0.645) )

            fleet[i].fatigued_out = yes;

        }
    }
    else if ( (fleet[i].tail_number == 708 || fleet[i].tail_number == 727)
        && (fleet[i].flei >= 0.733) )

            fleet[i].fatigued_out = yes;

        }
    }
}

//"regular" aircraft (aircraft with a FLEI limit = 1) that have fatigued out during the year

for (i=0; i < DIM_FLEET; i++)
{
    if ( (fleet[i].crashed == no) && (fleet[i].fatigued_out == no) )
    {
        if ( (fleet[i].epa == no) && (fleet[i].flei > 1) )
        {
```

```
                    fleet[i].fatigued_out = yes;
    ~
    else if ((fleet[i].epa == yes) && (fleet[i] flei > 0.333))
    ~
                    fleet[i] fatigued_out = yes;
    ~
    ~
~

//count the number of fatigued out aircraft up to the current year (cumulative)

nb_fatigued_out =0;
for (i=0; i < DIM_FLEET; i++)
{
    if (fleet[i].fatigued_out == yes)
    {
                    nb_fatigued_out = nb_fatigued_out + 1;
    ~
~

results_nb_fatigued_out[nb_it-1][year] = nb_fatigued_out;

//determine the number of remaining aircraft in the current year

remaining_ac = 0;
remaining_singles = 0;
remaining_duals = 0;

for (i=0; i < DIM_FLEET; i++)
{
    if ( (fleet[i].crashed == no) && (fleet[i] fatigued_out == no) )
    {
```

G-53

```
                    remaining_ac = remaining_ac + 1;
                    if (fleet[i].type == single)
                    {
                        remaining_singles = remaining_singles + 1;
                    }
                    else
                    {
                        remaining_duals = remaining_duals + 1;
                    }
                }

        results_remaining_ac[nb_it-1][year] = remaining_ac;
        results_remaining_singles[nb_it-1][year] = remaining_singles;
        results_remaining_duals[nb_it-1][year] = remaining_duals;
}

/*****************************************************************************
 *
 *           void  statistic_results()
 *
 * Using the data recorded in the "results" arrays, we calculate the mean values
 * of some of the parameters of interest, for each year.
 * Parameters of interest are, for example: the number of crashed aircraft,
 * the number of remaining aircraft, the number of aircraft waiting to go into
 * third line maintenance, etc.
 *
 * We are using the classical formula:
 *    - the mean is the arithmetic mean of the sample
 *
 ****************************************************************************/
```

```c
void statistic_results()
{
float sum=0. sum2=0;
float moyenne=0, variance=0, st_dev=0, moyenne_hours=0, monthly_average=0, annual_average=0;

//NUMBER OF REMAINING AIRCRAFT (creates a file formatted in such a way that it
//                              could be used as input file in DeltaGraph)

output = fopen("total.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_remaining_ac[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;  //calculate the mean

    //write data into the output file "total.txt"
    output = fopen("total.txt", "a+");
    fprintf(output, "%u", 2002+year);
    fprintf(output, "   %3.f\n", moyenne);
    fclose(output);
}

//NUMBER OF SINGLES REMAINING

output = fopen("singles.txt", "w");
fclose(output);
```

```
for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_remaining_singles[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;  //calculate the mean

    //write data into the output file "singles.txt"
    output = fopen("singles.txt", "a+");
    fprintf(output, "%u", 2002+year);
    fprintf(output, "   %3.f\n", moyenne);
    fclose(output);
}

//NUMBER OF DUALS REMAINING

output = fopen("duals.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_remaining_duals[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;  //calculate the mean

    //write data into the output file "duals.txt"
    output = fopen("duals.txt", "a+");
    fprintf(output, "%u", 2002+year);
    fprintf(output, "   %3.f\n", moyenne);
```

```
        fclose(output);

    };

//NUMBER OF CRASHES

output = fopen("crashes.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_crashes[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;  //calculate the mean

    //write data into the output file "crashes.txt"
    output = fopen("crashes.txt", "a+");
    fprintf(output, "%u", 2002+year);
    fprintf(output, "   %3.f\n", moyenne);
    fclose(output);

};

// NUMBER OF CRASHES

// this version creates a file formatted in such a way that it
// could be used as input file in DeltaGraph, allowing us to plot
// the number of crashes as a function of the number of hours flown
// by the fleet
```

G-57

```c
output = fopen("crashes_graph.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_crashes[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;  //calculate the mean

//modif for doing the graph nb. crashes function of hours flown

    sum2 = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum2 = sum2 + results_hours[nb_it-1][year];
    }
    moyenne_hours = sum2/NB_ITER;

    //write data into the output file "crashes_graph.txt"
    output = fopen("crashes_graph.txt", "a+");
    fprintf(output, "%6.f", moyenne_hours);
    fprintf(output, "   %3.f\n", moyenne);
    fclose(output);
}

//NUMBER OF FATIGUED-OUT AIRCRAFT

output = fopen("fatigued_ac.txt", "w");
fclose(output);
```

G-58

```c
for (year=0; year < MAX_YEARS; year++)
{
    sum = 0;
    for (nb_it=1; nb_it <= NB_ITER; nb_it++)
    {
        sum = sum + results_nb_fatigued_out[nb_it-1][year];
    }
    moyenne = sum/NB_ITER;   //calculate the mean

    //write data into the output file "duals.txt"
    output = fopen("fatigued_ac.txt", "a+");
    fprintf(output, "%u", 2002+year);
    fprintf(output, "  %3.f\n", moyenne);
    fclose(output);
}


//NUMBER OF AIRCRAFT IN SECOND LINE MAINTENANCE

output = fopen("second_line_maint.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    //write data into the output file "second_line_maint.txt"
    output = fopen("second_line_maint.txt", "a+");
    fprintf(output, "%u", 2001+year);

    for (month=1; month <= 12; month++)
    {
        sum = 0;
        for (nb_it=1, nb_it <= NB_ITER; nb_it++)
        {
            sum = sum + results_nb_second_line[nb_it-1][month-1][year];
```

```
			}
			monthly_average = sum/NB_ITER;

			fprintf(output, "  %3.f", monthly_average);

		}
		fprintf (output, "\n");
		fclose(output);

	}

//NUMBER OF AIRCRAFT IN THIRD LINE MAINTENANCE

output = fopen("third_line_maint.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{

	//write data into the output file "third_line_maint.txt"
	output = fopen("third_line_maint.txt", "a+");
	fprintf(output, "%u", 2001+year);

	for (month=1; month <= 12; month++)
	{

		sum = 0;
		for (nb_it=1; nb_it <= NB_ITER; nb_it++)
		{

			sum = sum + results_nb_third_line[nb_it-1][month-1][year];

		}
		monthly_average = sum/NB_ITER;
		fprintf(output, "  %3.f", monthly_average);

	}
	fprintf (output, "\n");
	fclose(output);

}
```

```
//NUMBER OF AIRCRAFT WAITING TO GO INTO THIRD LINE MAINTENANCE

output = fopen("waiting.txt", "w");
fclose(output);

for (year=0; year < MAX_YEARS; year++)
{
    //write data into the output file "waiting.txt"
    output = fopen("waiting.txt", "a+");
    fprintf(output, "%u", 2001+year);

    for (month=1; month <= 12; month++)
    {
        sum = 0;
        for (nb_it=1; nb_it <= NB_ITER; nb_it++)
        {
            sum = sum + results_nb_waiting[nb_it-1][month-1][year];
        }
        monthly_average = sum/NB_ITER;

        fprintf(output, " %3.f", monthly_average);
    }
    fprintf (output, "\n");
    fclose(output);
}

//NUMBER OF OPERATIONAL AIRCRAFT (AIRCRAFT THAT ARE AVAILABLE FOR MISSIONS)
//ON A MONTHLY BASIS

output = fopen("operational.txt", "w");
```

G-61

```c
    fclose(output);

    for (year=0; year < MAX_YEARS; year++)
    {
        output = fopen("operational.txt", "a+");
        fprintf(output, "%u", 2001+year);

        for (month=1; month <= 12; month++)
        {
            sum = 0;
            for (nb_it=1; nb_it <= NB_ITER; nb_it++)
            {
                sum = sum + results_nb_operational[nb_it-1][month-1][year];
            }

            monthly_average = sum/NB_ITER;

            fprintf(output, " %3.f", monthly_average);
        }
        fprintf (output, "\n");
        fclose(output);
    }
}

//NUMBER OF OPERATIONAL AIRCRAFT ON A YEARLY BASIS

    output = fopen("operational_yearly.txt", "w");
    fclose(output);

    for (year=0; year < MAX_YEARS; year++)
    {
        sum = 0;
        for (month = 1; month <= 12; month++)
        {
```

```
        for (nb_it=1; nb_it <= NB_ITER; nb_it++)
        {
            sum = sum + results_nb_operational[nb_it-1][month-1][year];
        }
    }

    moyenne = sum/(12 * NB_ITER);  //calculate the mean

    //write data into the output file "operational_yearly.txt"
    output = fopen("operational_yearly.txt", "a+");
    fprintf(output, "%u", 2001+year);
    fprintf(output, "   %3.f\n", moyenne);
    fclose(output);

}

system("type total.txt");  //displays the "total.txt" file on the screen

void CCF18forecastingDlg::OnCancelButton()
{
    // TODO: Add your control notification handler code here

    /////////////////////////////
    // MY CODE STARTS HERE
    /////////////////////////////

    OnOK();

    /////////////////////////////
    // MY CODE ENDS HERE
    /////////////////////////////

}
```

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1 ORIGINATOR (the name and address of the organization preparing the document Organizations for whom the document was prepared e g Establishment Sponsoring a contractor's report, or tasking agency, are entered in Section 8) OPERATIONAL RESEARCH DIVISION DEPARTMENT OF NATIONAL DEFENCE OTTAWA, ONTARIO K1A 0K2	2 SECURITY CLASSIFICATION (overall security classification of the document, including special warning terms if applicable) UNCLASSIFIED

3 TITLE (the complete document title as indicated on the title page Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title)

CF-18 STRUCTURAL LIFE ANALYSIS MODEL: A USER'S GUIDE

4 AUTHORS (last name, first name, middle initial)

STEMATE, LUMINITA C.

5 DATE OF PUBLICATION (month Year of Publication of document) JUNE 2001	6a NO OF PAGES (total containing information Include Annexes, Appendices, etc) 129	6b NO OF REFS (total cited in document) 8

7 DESCRIPTIVE NOTES (the category of document, e g, technical report, technical note or memorandum If appropriate, enter the type of report e g interim, progress, summary, annual or final Give the inclusive dates when a specific reporting period is covered)

PROJECT REPORT

8 SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development Include the address)

CHIEF OF THE AIR STAFF

9a PROJECT OR GRANT NO (if appropriate, the applicable research and development project or grant number under which the document was written Please specify whether project or grant)	9b. CONTRACT NO (if appropriate, the applicable number under which the document was written)

10a ORIGINATOR's document number (the official document number by which the document is identified by the originating activity This number must be unique to this document) ORD PROJECT REPORT PR 2001/08	10b OTHER DOCUMENT NOS (Any other numbers which may be assigned this document either by the originator or by the sponsor)

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

  (X) Unlimited distribution
  ( ) Distribution limited to defence departments and defence contractors further distribution only as approved
  ( ) Distribution limited to defence departments and Canadian defence contractors, further distribution only as approved
  ( ) Distribution limited to government departments and agencies, further distribution only as approved
  ( ) Distribution limited to defence departments, further distribution only as approved
  ( ) Other (please specify)

12 DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in 11) is possible, a wider announcement audience may be selected)

13 ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself It is highly desirable that the abstract of classified documents be unclassified Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), or (U) It is not necessary to include here abstracts in both official languages unless the test is bilingual)

This study represents a comprehensive *user's guide* for the 2000 version of the CF-18 Structural Life Analysis Model (CF-18 SLAM) that was originally developed in 1989 and has been used ever since as a support tool for various decisions regarding the structural life of the CF-18 fleet. This guide offers a high-level, "philosophical" perspective on the model, as well as a low-level, programming perspective, the whole complemented by a suite of discussed examples.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document They should be selected so that no security classification is required Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included If possible keywords should be selected from a published thesaurus, e g Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified . If it is not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title)

CF-18
SIMULATION
MODEL
FORECASTING
FATIGUE
ATTRITION
STRUCTURAL LIFE
AIRCRAFT MAINTENANCE
FIGHTERS
USER'S GUIDE

Canada

#515910